

**POINT PROCESS-BASED MODELING AND ANALYSIS OF ASYNCHRONOUS
EVENT SEQUENCES**

A Dissertation
Presented to
The Academic Faculty

By

Hongteng Xu

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology

August 2017

Copyright © Hongteng Xu 2017

POINT PROCESS-BASED MODELING AND ANALYSIS OF ASYNCHRONOUS EVENT SEQUENCES

Approved by:

Dr. Hongyuan Zha, Advisor
School of Computational Science
and Engineering
Georgia Institute of Technology

Dr. Mark A. Davenport, co-advisor
School of Electrical and Computer
Engineering
Georgia Institute of Technology

Dr. Le Song
School of Computational Science
and Engineering
Georgia Institute of Technology

Dr. Justin K. Romberg
School of Electrical and Computer
Engineering
Georgia Institute of Technology

Dr. Chuanyi Ji
School of Electrical and Computer
Engineering
Georgia Institute of Technology

Dr. Bistra Dilkina
School of Computational Science
and Engineering
Georgia Institute of Technology

Date Approved: June 2, 2017

To my parents:

Xianqi Xu and Rongqing Wang

and to my wife:

Dixin Luo

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my Ph.D. advisors Dr. Hongyuan Zha and Dr. Mark A. Davenport for providing me an opportunity to work in their labs and continuously supporting my study and related research. Their inspiring thought, insightful guidance, and kind encouragement help me move to the right track during these years. Without their supervision, I could not achieve what I have currently.

My research achievements also benefit significantly from interacting with a marvelous group of colleagues and collaborators. I would like to extend my sincere appreciation to each of them: Xia Ning, Shamim Nemati, Weiyao Lin, Ren-Cang Li, Yi Xu, Junchi Yan, Yi Zhen, Mehrdad Farajtabar, Licheng Yu, etc. In particular, I am grateful to Dr. Xia Ning for her generous help about career and endeavor, and to Dr. Shamim Nemati, Dr. Weiyao Lin for bringing me to many new fields. I would like to thank Dr. Junchi Yan, Dr. Yi Zhen, and Dr. Yi Xu for active discussions and close collaborations. I also thank Mehrdad and Licheng for the timely support whenever I need help.

Additionally, I would like to thank the other members of my Ph.D. dissertation committee: Dr. Justin K. Romberg, Dr. Le Song, Dr. Chuanyi Ji and Dr. Bistra Dilkina, for their time to attend my thesis defense and their insightful suggestions on my dissertation, which are of great help. I would also like to thank all my friends from Georgia Tech Shanghai program for encouraging and accompanying me.

Last but not the least, I feel deeply indebted to the selfless love and endless support of my parents and my parents in law. My wife, Dixin, you are not only the most significant soulmate in my life but also a great helper in my research work. I always thank you for believing me unconditionally. You and our parents are always the most important part of my life.

TABLE OF CONTENTS

| | |
|--|-----|
| Acknowledgments | v |
| List of Tables | xii |
| List of Figures | xiv |
| Chapter 1: Introduction | 1 |
| 1.1 Motivations | 1 |
| 1.1.1 Asynchronous Event Sequences | 2 |
| 1.1.2 Temporal Point Processes | 3 |
| 1.2 Research Problems | 4 |
| 1.3 Literature Survey | 7 |
| 1.3.1 Continuous-time Models | 7 |
| 1.3.2 Typical Point Processes | 8 |
| 1.3.3 Causality Analysis | 10 |
| 1.3.4 Clustering Analysis | 10 |
| 1.3.5 Learning from Imperfect Observations | 11 |
| 1.4 Contributions and Organizations | 12 |
| Chapter 2: Learning Granger Causality Based on Hawkes Processes | 15 |

| | | |
|-------|--|----|
| 2.1 | Introduction | 15 |
| 2.2 | Granger Causality Based on Point Processes | 17 |
| 2.3 | Proposed Model | 18 |
| 2.3.1 | Granger Causality Based on Time-invariant Hawkes Process | 18 |
| 2.3.2 | Learning Task | 19 |
| 2.4 | Learning Algorithm | 22 |
| 2.4.1 | An EM-based Algorithm | 22 |
| 2.4.2 | Adaptive Selection of Basis Functions | 25 |
| 2.4.3 | Properties of The Proposed Method | 27 |
| 2.5 | Experiments | 28 |
| 2.5.1 | Synthetic Data | 29 |
| 2.5.2 | Real-world Data | 34 |
| 2.6 | Summary | 36 |

Chapter 3: Feature-based Clustering: Multi-Task Multi-Dimensional Hawkes Processes

| | | |
|-------|---|----|
| 3.1 | Introduction | 37 |
| 3.2 | Proposed Model | 38 |
| 3.2.1 | Multi-Task Multi-Dimensional Hawkes Processes | 38 |
| 3.2.2 | Structural Constraints on MMHP | 39 |
| 3.2.3 | Learning Task | 39 |
| 3.3 | Learning Algorithm | 40 |
| 3.3.1 | Learning MMHP | 40 |
| 3.3.2 | Clustering Based on Infectivity Tensor | 43 |

| | | |
|---|---|-----------|
| 3.4 | Experimental Results | 44 |
| 3.4.1 | Synthetic Data | 46 |
| 3.4.2 | Real-world Data | 48 |
| 3.5 | Summary | 50 |
| Chapter 4: Model-based Clustering: Dirichlet Mixture Model of Hawkes Process | | 51 |
| 4.1 | Introduction | 51 |
| 4.2 | Proposed Model and Learning Algorithm | 53 |
| 4.2.1 | Dirichlet Mixture Model of Hawkes Processes | 53 |
| 4.2.2 | Variational Bayesian Learning Algorithm | 55 |
| 4.2.3 | Adaptive Inner-Outer Iteration Strategies | 58 |
| 4.3 | Further Analysis | 62 |
| 4.3.1 | Local Identifiability | 62 |
| 4.3.2 | Computational Complexity and Acceleration | 65 |
| 4.4 | Experiments | 66 |
| 4.4.1 | Synthetic Data | 67 |
| 4.4.2 | Real-world Data | 69 |
| 4.5 | Summary | 72 |
| Chapter 5: Learning Hawkes Processes from Short Doubly-Censored Sequences | | 74 |
| 5.1 | Introduction | 74 |
| 5.2 | Learning from SDC Event Sequences | 76 |
| 5.2.1 | Data Synthesis via Sampling-Stitching | 76 |
| 5.2.2 | Justification | 78 |

| | | |
|--|--|-----------|
| 5.2.3 | Feasibility | 80 |
| 5.3 | Implementation for Time-varying Hawkes Processes | 81 |
| 5.4 | Experiments | 85 |
| 5.4.1 | Implementation Details | 85 |
| 5.4.2 | Synthetic Data | 85 |
| 5.4.3 | Real-World Data | 88 |
| 5.5 | Summary | 93 |
| Chapter 6: Discriminative Learning of Point Processes and Patient Flow Prediction | | 94 |
| 6.1 | Introduction | 94 |
| 6.1.1 | Motivations | 94 |
| 6.1.2 | A Typical Application: Patient Flow Prediction | 95 |
| 6.2 | Problem Statement and Data Analysis | 98 |
| 6.2.1 | Notations and Problem Statement | 98 |
| 6.2.2 | Data and Basic Statistics | 99 |
| 6.3 | Proposed Method | 101 |
| 6.3.1 | Mutually-Correcting Processes | 101 |
| 6.3.2 | Discriminative Learning Algorithm | 105 |
| 6.3.3 | Enhancing Robustness to Imbalanced Data | 109 |
| 6.3.4 | Patient Flow Prediction | 112 |
| 6.3.5 | The Connection with Deep Learning | 112 |
| 6.4 | Experiments | 114 |
| 6.4.1 | Baselines and Evaluations | 114 |

| | | |
|--|--|------------|
| 6.4.2 | Comparison Results | 119 |
| 6.4.3 | Feature Selection Result | 122 |
| 6.5 | Summary | 123 |
| Chapter 7: Point Process-based Trailer Generation | | 125 |
| 7.1 | Introduction | 125 |
| 7.1.1 | Motivations | 125 |
| 7.1.2 | Background | 127 |
| 7.2 | Properties of Movie Trailer | 127 |
| 7.2.1 | Measure and Dynamics of Attractiveness | 128 |
| 7.2.2 | The Dynamics of Music | 130 |
| 7.3 | Point Process-based Attractiveness Model | 131 |
| 7.3.1 | Motivation | 131 |
| 7.3.2 | Self-Correcting Point Process | 132 |
| 7.3.3 | Model Learning | 133 |
| 7.4 | Trailer Generation | 135 |
| 7.5 | Experiments | 138 |
| 7.5.1 | Objective Evaluation | 139 |
| 7.5.2 | Subjective Evaluation | 141 |
| 7.6 | Summary | 142 |
| Chapter 8: Conclusions | | 143 |
| Appendix A: Details of Basis Function Selection | | 147 |

| | |
|---|-----|
| Appendix B: Implementation Details of DMHP | 150 |
| B.1 Nested EM Framework | 150 |
| B.2 Update The Number of Clusters K via MCMC | 153 |
| Appendix C: The Proof of Local Identifiability of DMHP | 155 |
| References | 170 |
| Vita | 171 |

LIST OF TABLES

| | | |
|-----|--|-----|
| 1.1 | Contributions and Organizations of This Dissertation | 14 |
| 2.1 | Comparison for various methods | 28 |
| 2.2 | The <i>Loglike</i> ($\times 10^6$) obtained via various methods | 35 |
| 3.1 | Clustering on Synthetic Data (<i>ClusAcc</i> ($\times 100\%$)). | 48 |
| 3.2 | Clustering Performance on IPTV Data ($\times 10^7$). | 50 |
| 4.1 | The clustering purity on the synthetic data with sine-like impact functions. . | 67 |
| 4.2 | The clustering purity on the synthetic data with piecewise constant impact functions. | 67 |
| 4.3 | The clustering consistency on real-world data sets. | 72 |
| 6.1 | Number of patients and transitions, and average durations (days) in each CU. | 101 |
| 6.2 | The proportions of nonzero elements in different feature domains in each CU. | 101 |
| 6.3 | Comparison of various parametric point processes. | 103 |
| 6.4 | Prediction accuracy for various methods on destination CUs | 117 |
| 6.5 | Prediction accuracy for various methods on duration days | 119 |
| 6.6 | Overall prediction accuracy for various methods on relative simulation errors | 120 |
| 7.1 | The statistics of normalized fixation variance ($\times 10^8$) | 129 |

| | | |
|-----|---|-----|
| 7.2 | Comparison on computational cost. | 140 |
|-----|---|-----|

LIST OF FIGURES

| | | |
|-----|---|----|
| 1.1 | Illustration of conditional intensity functions. | 3 |
| 1.2 | The four research problems studied in our work. | 5 |
| 2.1 | In the left subfigure, an event sequence generated via a 3-dimensional Hawkes process is given, where the event with red type is only Granger caused via historical events with blue type. In the right subfigure, the group sparsity of learned impact functions indicate the structure of the Granger causality graph. | 19 |
| 2.2 | Each vector in the tensor of parameters represents the coefficients of an impact function. | 20 |
| 2.3 | The curves of <i>Loglike</i> w.r.t. the change of α_P , α_G and α_S are shown. In each subfigure, left: $\alpha_G = 100$, $\alpha_S = 10$, $\alpha_P \in [10^{-2}, 10^4]$; middle: $\alpha_G = 100$, $\alpha_P = 1000$, $\alpha_S \in [10^{-2}, 10^4]$; right: $\alpha_P = 1000$, $\alpha_S = 10$, $\alpha_G \in [10^{-2}, 10^4]$ | 30 |
| 2.4 | The e_μ , e_ϕ , and <i>Loglike</i> for various methods. | 31 |
| 2.5 | Contributions of regularizers: the comparisons of the impact functions obtained via MLE-SGLP and pure MLE. The green subfigures represent the all-zero impact functions. The black curves are real impact functions. The blue curves are the estimates based on pure MLE, and the red curves are the estimates based on our MLE-SGLP. | 32 |
| 2.6 | (a) The infectivity matrix of TV programs. The element in the c -th row and the c' -th column is $\int_0^\infty \phi_{cc'}(s)ds$. (b) The estimates of the nonzero impact functions learned from the IPTV data. By ranking the infectivity $\int_0^\infty \phi_{cc'}(s)ds$ from high to low, the top 24 impact functions are shown. For the convenience of visualization, $\phi_{cc'}^{0.25}(t)$ is shown in each subfigure. | 33 |
| 3.1 | Parameter study for the MMHP. | 46 |

| | | |
|-----|---|----|
| 3.2 | Experimental results on synthetic data. | 48 |
| 3.3 | Experimental results on an IPTV system. | 49 |
| 4.1 | Comparison for various inner-outer iteration strategies on different synthetic data sets. The testing data set contains 2-5 clusters, and each cluster contains 200 event sequences generated via a 5-dimensional Hawkes process. Each convergence curve is the average of 5 trials' results. For the fairness, 20 outer iterations are applied in each trial. For heuristic strategy, the increasing method increases the number of inner iterations from 2 to 8; the decreasing method decreases the number from 8 to 2. The constant method fixes the number to 5. | 60 |
| 4.2 | For each method, the ground truth of responsibility is shown as a black line, the responsibility after 15 inner iterations is shown as red dot, and its mean is shown as red line. Using increasing strategy, the responsibility reaches the ground truth more quickly. | 60 |
| 4.3 | Comparison for various methods on F1 score of minor cluster with various number of events per sequence. | 64 |
| 4.4 | Histograms of clusters' numbers. | 69 |
| 4.5 | Comparison on infectivity matrices of ICU patient data. | 70 |
| 4.6 | Comparison on infectivity matrices of IPTV data. | 71 |
| 5.1 | Illustration of a time-varying Hawkes processes, which can be used to model the dynamics of disease network over the age of patient. | 75 |
| 5.2 | The illustration of our sampling-stitching process. | 78 |
| 5.3 | Comparisons on log-likelihood and relative error in the case of time-invariant Hawkes processes. | 86 |
| 5.4 | Comparisons on log-likelihood and relative error in the case of time-varying Hawkes processes. | 87 |
| 5.5 | Comparisons on infectivity $\{\psi_{cc'}(t)\}$ in the case of time-varying Hawkes processes. The number of original SDC sequences is 200 and stitched via our method once. | 88 |
| 5.6 | Comparisons on the log-likelihood of the testing data. | 90 |

| | | |
|-----|--|-----|
| 5.7 | Comparisons on the infectivity $\{\psi_{cc'}(t)\}$ w.r.t. the age. | 91 |
| 5.8 | For each SDC sequence, i.e., incomplete disease history of a person in his lifetime, we design a mechanism to select other SDC sequences as predecessors/successors and synthesize a long sequence. Then, we can estimate the unobserved triggering patterns among diseases, i.e., the red dashed arrows, and construct a disease network. | 91 |
| 5.9 | The network of circulatory system diseases in different ages. The diseases (nodes) are labeled with ICD-9 codes. The diseases belonging to different sub-categories are labeled with different colors. The size of the c -th node is $\sum_{c'} \psi_{cc'}(t)$, which reflects the importance of the c -th disease. The width of directed edge is set as $\psi_{cc'}(t)$, which indicates the strength of triggering pattern. | 92 |
| 6.1 | (a) The transition process of an old male patient having coronary heart disease may include the Coronary Care Unit for preoperative tests, the Anesthesia Services for cardiac surgery, the Cardiac Surgery Recovery unit, and finally the Medical ICU and general ward for nursing. The transition process of a pregnant woman having a premature baby may include the Anesthesia Services for a Caesarean section surgery, the Medical Care Unit for the mother, and the Neonatal Care Unit for the baby. There are overlaps between the elderly patient and the pregnant woman in the Anesthesia ICU and the Medical ICU. (b) The destination CUs and the duration days are represented via two event sequences. Along the time line, the color dots indicate various CUs and the color lines with various length indicate the durations (in units of days). Applying our mutually-correcting process model, the conditional intensity functions for CUs and durations are proposed to capture the positive and negative influences among unit types and durations, respectively. | 97 |
| 6.2 | Comparison on conditional intensity function for various point processes. An event sequence is given and the conditional intensity functions of various point processes are shown. | 103 |
| 6.3 | The simple 2-D example illustrating various methods to solve data imbalance problem: (a) The original data having 3 classes is shown, where blue crosses are samples of major class while red squares and green dots are samples of two minor classes, respectively. (b) The weighed data is shown, where the samples of minor classes have large weights (enlarged). (c) The hierarchical data is shown, where the samples are unchanged while a non-linear binary classifier is learned. (b) The synthetic data is shown, where the minor classes are supplemented via auxiliary samples. The classifiers in the subfigures are shown as black dotted lines and curves. | 110 |

| | | |
|-----|---|-----|
| 6.4 | The discriminative learning of point processes implemented via a neural network. | 113 |
| 6.5 | (a) The prediction accuracy of each CUs and the overall accuracy are given. In each subfigure, the color bars correspond to various learning methods. (b) The prediction accuracy of each duration day and the overall accuracy are given. In each subfigure, the color bars correspond to various learning methods. | 118 |
| 6.6 | The relative simulation error of each CU and the overall simulation error are given. In each subfigure, the color bars correspond to various learning methods. | 119 |
| 6.7 | Feature selection result. | 123 |
| 7.1 | The left subfigure shows four shots in trailer “ <i>The Wolverine 2013</i> ” and their fixation variances. The right subfigure shows dynamics of fixation variance calculated from training trailers. | 129 |
| 7.2 | The Saliency Points of Music v.s. Montage Positions (Trailer of “ <i>The Bling Ring</i> ”). | 131 |
| 7.3 | The scheme of our trailer generator. | 138 |
| 7.4 | The mean, the median and the standard deviation of the fixation variance σ for various methods. | 140 |
| 7.5 | The box plots of scores for various methods on three questions. The red crosses are means and the red bars are medians. | 141 |

SUMMARY

Point processes have been widely used to model complicated event sequences produced from natural and social systems. In this work, we study the problems of asynchronous event sequence analysis with the help of point process-based modeling and learning techniques, and establish a unified algorithmic framework to learn point processes and capture the latent information behind seemingly “random” data. In particular, with this common framework, we make efforts to solve the following four problems:

- **Causality Analysis of Asynchronous Event Sequences.** Learning Granger causality from asynchronous event sequences for general point processes is a very challenging task. We propose an effective approach to learn Granger causality for a special but significant type of point processes — the Hawkes process. According to the relationship between a Hawkes process’s impact functions and its Granger causality graph, our model represents the impact functions using a series of basis functions and recovers the Granger causality graph via group sparsity of the impact functions’ coefficients. The strategy of selecting basis functions is discussed in detail. In order to enhance the robustness, we propose an effective learning algorithm combining a maximum likelihood estimator (MLE) with a sparse-group-lasso (SGL) regularizer. Additionally, the flexibility of our model allows us to incorporate the clustering structure of event types into learning framework.
- **Clustering Analysis of Asynchronous Event Sequences.** We study the problem of clustering asynchronous event sequences in depth and propose two clustering strategies based on Hawkes processes. The first is a feature-based strategy, which extracts features from event sequences and then learns clusters based on the features. In the phase of feature extraction, we propose a multi-task multi-dimensional Hawkes process model and learn adjacent matrices of the Granger causality graphs for the event sequences with the help of multi-task learning. The sparse and low-

rank structures of the tensor corresponding to those adjacent matrices are considered in our learning algorithm. The second is a model-based strategy, which establishes a mixture model directly based on Hawkes processes. We propose a Dirichlet mixture model of Hawkes processes, and design a variational inference algorithm in a nested expectation-maximization (EM) framework to learn the model effectively. We compare these two strategies on their identifiability, convergence, computational complexity, clustering purity, and clustering consistency, and demonstrate their potentials in practical applications, e.g., IPTV user clustering, and patient clustering.

- Robust Learning of Point Processes.** How to learn point processes from incomplete observations is a common and challenging problem in many real-world applications. We study this critical problem in two perspectives. On one hand, from the viewpoint of data, we propose a data synthesis method to generate long event sequences from original short doubly-censored ones via a sampling-stitching mechanism. The rationality and the feasibility of our data synthesis method are discussed in details. On the other hand, from the viewpoint of learning algorithm, we propose a discriminative learning framework for learning point processes, which can be explained as an implementation of neural networks. Compared with the generative learning method, our discriminative learning method requires fewer samples, and achieves encouraging results for predicting future events. We achieve robust learning of many complicated point processes (e.g., mutually-correcting point processes and time-varying Hawkes processes) and apply them to analyze admission records of patients.
- Innovative Applications of Point Processes.** Point processes have attracted a lot of researchers in the field of data analysis and data mining. However, few researchers consider the applications of point processes on computer vision. Driven by the problem of movie trailer generation, we propose a self-correcting process model to measure the visual attractiveness of movie clips. The loss of attractiveness is used as a

criterion to select and re-organize attractive clips. To the best of our knowledge, this is a pioneering work in the use of point processes to achieve video analysis.

Keywords: Point process, Hawkes process, correcting process, Granger causality, impact function, infectivity network, multi-task learning, Dirichlet mixture model, structural regularizer, discriminative learning, doubly-censored data, attractiveness model.

CHAPTER 1

INTRODUCTION

1.1 Motivations

With the fast development of modern communications, networks, and mobile devices, real-world interactions among multiple entities in information systems constantly produce a large amount of event data, and the relationships among the entities are hidden in the data. We take some typical examples from diverse domains to demonstrate the pervasiveness of these data.

- **Social networks.** In social networks like facebook and twitter, we post our daily experience and share other people's information. In particular, our social behaviors (i.e., posting information and responding others) are likely to trigger other people's feedback. These interactions reveal the infectivity among different users.
- **Job markets.** On the job market, employees search different jobs and hop among different companies. Competitive companies can attract new employees consistently and keep their existing employees from hopping while the companies with weak competitiveness are likely to suffer to talent crisis. The flow of talents among different companies reflects the relationships of the companies on the job market.
- **IPTV systems.** In IPTV systems, users' viewing behaviors (i.e., when and which programs they watched) are recorded, which reveal their viewing patterns and preferences. After watching an actor's news, we are likely to watch his movies and dramas in the future. After watching an episode drama at certain time, we are likely to watch the remaining episodes daily or weekly in the following days. The triggering patterns among TV programs can be viewed as features of users' viewing preferences.

- **Electronic health records.** Electronic health records contain the information of diseases' diagnoses and treatments. The progress of a disease and its influence on other diseases can be captured via the historical admissions of different patients, and a disease network can be constructed accordingly.

All these event data are formulated as asynchronous event sequences, which contain a series of time stamps, event marks, and optionally, high-dimensional features. The dynamics and the interactions of the events reflect the relationships among entities hidden in the randomness of data. The focus of this dissertation is on establishing a unified point process-based framework for asynchronous event sequence modeling and analysis and developing flexible models and robust learning algorithms for various challenging problems and interesting applications.

1.1.1 Asynchronous Event Sequences

Typically, an asynchronous event sequence can be represented as $\mathbf{s} = \{(t_i, c_i)\}_{i=1}^I$, where $t_i \in [T_b, T_e]$ and $c_i \in \mathcal{C}$. Here T_b and T_e represent the beginning and the ending of the observation window. $\mathcal{C} = \{1, \dots, C\}$ represents the set of event types. A point process with C event types is represented by C counting processes $\{N_c\}_{c=1}^C$ on a probability space $(\Omega, \mathfrak{F}, \mathbb{P})$ [1]. $N_c = \{N_c(t) | t \in [T_b, T_e]\}$, where $N_c(t)$ is the number of type- c events occurring at or before time t . $\Omega = [T_b, T_e] \times \mathcal{C}$ is the sample space. $\mathfrak{F} = (\mathfrak{F}(t))_{t \in \mathbb{R}}$ is the filtration representing the set of event sequences the process can realize until time t . Finally, \mathbb{P} denotes the probability measure.

Different from traditional time series and synchronous (or aggregated) data, asynchronous event sequences contain discrete events randomly distributed in the continuous time domain. In this case, both the order of events and the time intervals between them are important for describing the dynamics of sequences, which increases the difficulty of modeling. In other words, focusing on asynchronous event sequences, we need to establish an effective model that captures not only the types of current and future events but also the

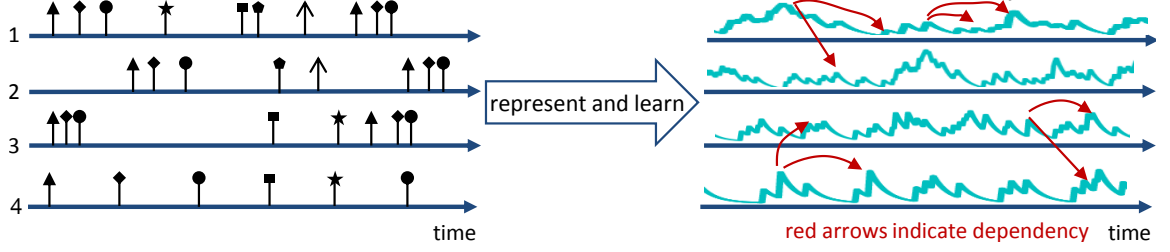


Figure 1.1: Illustration of conditional intensity functions.

happening time of them. Fortunately, the theory of temporal point processes provides us with an useful tool for achieving this aim.

1.1.2 Temporal Point Processes

A point process can be characterized via its conditional intensity functions $\{\lambda_c(t)\}_{c=1}^C$, where $\lambda_c(t)$ represents the expected instantaneous happening rate of type- c events given historical record:

$$\begin{aligned}
 \lambda_c(t) &= \lambda_c(t \mid \mathfrak{F}(t)) \\
 &= \frac{\mathbb{E}[dN_c(t) \mid \mathcal{H}_t^C]}{dt} \\
 &= \frac{P(c\text{-type event in } [t, t + dt) \mid \mathcal{H}_t^C)}{dt} \\
 &= \frac{P(c\text{-type event in } [t, t + dt) \mid \text{no event in } [t_i, t), \mathcal{H}_t^C)}{dt} \\
 &= \frac{P(c\text{-type event in } [t, t + dt), \text{ and no event in } [t_i, t) \mid \mathcal{H}_t^C)}{P(\text{no event in } [t_i, t) \mid \mathcal{H}_t^C)dt} \\
 &= \frac{p(t, c)}{1 - P(t)}.
 \end{aligned} \tag{1.1}$$

Here, $\mathcal{H}_t^C = \{(t_i, c_i) \mid t_i < t, c_i \in \mathcal{C}\}$ is the set of history, t_i is the time stamp of the last event before t . $p(t, c)$ is the conditional density of c -type event at time t given history, and $P(t)$ is the conditional cumulative distribution that there are some events happening in the interval $(t_i, t]$.

The conditional intensity function actually provides us with an effective representation of asynchronous event sequences — with the help of conditional intensity function, we

can represent each event sequence as a or a set of continuous functions of time, as Fig. 1.1 shows. It is beneficial for us to model the dynamics of the sequences. Moreover, given conditional intensity function, we can learn the corresponding point process using traditional statistical learning techniques. In particular, based on the definition of conditional intensity functions in (1.1), we have

$$p(t, c) = \lambda_c(t) \exp \left(- \sum_{c'=1}^C \int_{t_i}^t \lambda_{c'}(s) ds \right), \quad (1.2)$$

and

$$P(t) = 1 - \exp \left(- \sum_{c'=1}^C \int_{t_i}^t \lambda_{c'}(s) ds \right). \quad (1.3)$$

Therefore, given an event sequence $\mathbf{s} = \{(t_i, c_i)\}_{i=1}^I$, its likelihood can be calculated as:

$$\begin{aligned} \mathcal{L}(\mathbf{s}; \Theta) &= \prod_{i=1}^I p(t_i, c_i) \times (1 - P(T_e)) \\ &= \prod_{i=1}^I \lambda_{c_i}(t_i) \times \exp \left(- \sum_{c=1}^C \int_{T_b}^{T_e} \lambda_c(s) ds \right), \end{aligned} \quad (1.4)$$

where Θ is the parameter of the point process model.

1.2 Research Problems

The main interest of our research lies in modeling asynchronous event sequences based on point processes and analyzing the triggering pattern of the events and the relationship of the event sequences quantitatively. In particular, we study four challenging research problems and explore their potential solutions throughout the thesis.

Causality Analysis of Asynchronous Event Sequences. Asynchronous event sequences with multiple event types can always be modeled via multi-dimensional point processes in which each dimension corresponds to an event type. Generally, historical events in a sequence have influences on the appearance of current (and future) events, as the red

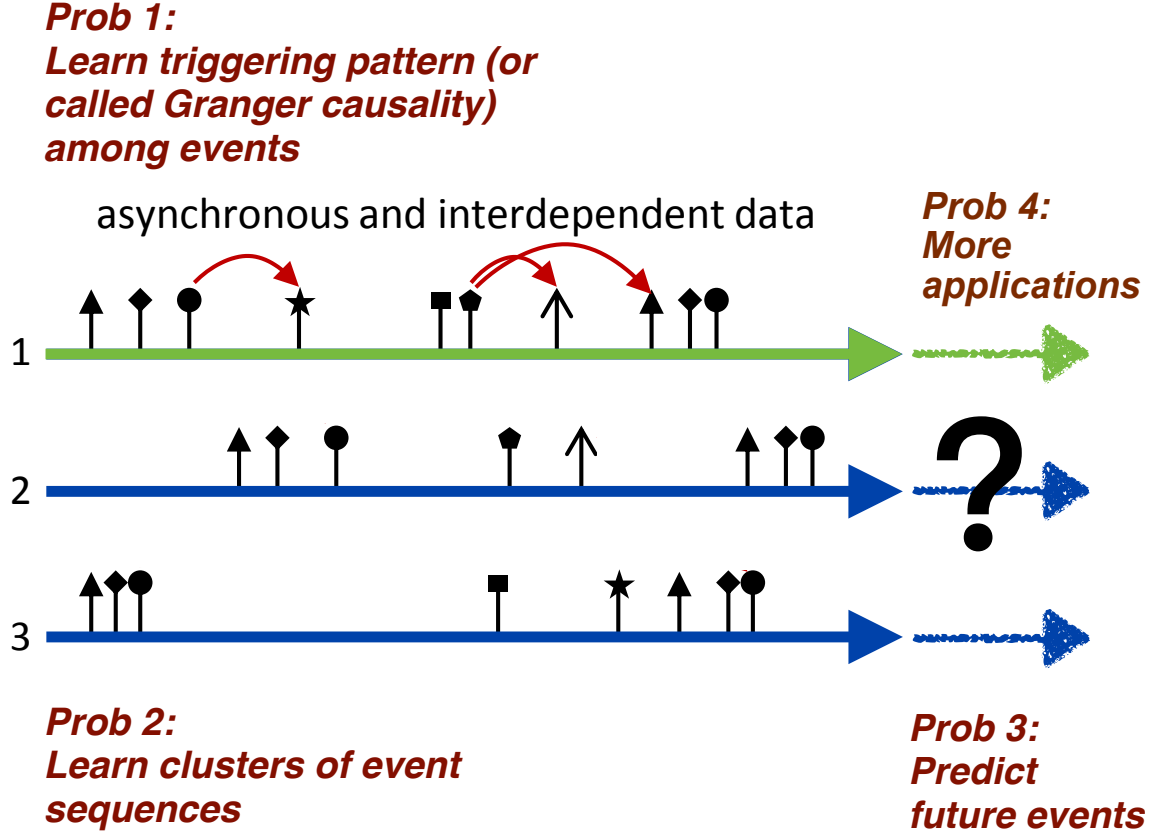


Figure 1.2: The four research problems studied in our work.

arrows in Fig. 1.2. In order to analyze the influences quantitatively, we need to learn the triggering pattern among different event types. From the viewpoint of graphical models, this means constructing a directed graph called Granger causality graph (or local independence graph) [2] based on the event sequences, in which each node is an event type and each arrow connecting two nodes indicates that the appearance of the event corresponding to the destination node is dependent on the historical events corresponding to the source node.

This task is very challenging — it is hard to find a universal and tractable representation of the complicated historical events to describe Granger causality for asynchronous event sequences. Because asynchronous event sequences are in the continuous time domain and no fixed time-lagged observation is available, traditional learning methods focusing on time series [3, 4, 5, 6] do not work in our case. A potential solution is constructing features for

various dimensions from historical events and learning Granger causality via a feature selection algorithm [7]. However, this method is highly dependent on the specific feature construction method used, resulting in dubious Granger causality. Therefore, the first problem we want to solve is “How to learn Granger causality for asynchronous event sequences robustly?” We devote Chapter 2 to the solution of the problem with solid theoretical analysis and convincing experiments.

Clustering Analysis of Asynchronous Event Sequences. It is common in practice that the asynchronous event sequences we observe exhibit clustering structure. Specifically, when we model these event sequences based on point processes, it is likely that they are generated via different point process models (i.e., the sequences with different colors in Fig. 1.2). An important task for us is learning their clustering structure in an unsupervised way. Similar to the causality analysis mentioned above, although many clustering methods have been proposed for time series [8, 9, 10], few researchers consider the clustering problem of asynchronous event sequences because of their asynchronous property.

Focusing on this challenging problem, we would like to answer the following question “Can we propose a robust and scalable clustering method for event sequences?” We can explore the answer of this question with the help of two strategies: 1) proposing a feature-based method and decomposing the clustering problem into two subproblems — extracting features and learning clusters; 2) proposing a model-based method, e.g., designing a mixture model for event sequences. We discuss the advantages and the disadvantages of these two strategies and give theoretical analysis and empirical experimental results in Chapters 3 and 4

Robust and Predictive Learning of Point Processes. Learning point processes requires us to collect a large amount of asynchronous event sequences as training data. However, in many real-world applications only incomplete or extremely short observations are available, which leads to the failure of learning algorithms. In this thesis, we make attempts to study the problem that “How to learn point processes from imperfect observations”.

To achieve robust and predictive learning of point processes, it is necessary to think in two directions — improving the quality of the training data and enhancing the robustness of the learning algorithms. In terms of data, we can introduce data synthesis methods as the pre-processing of training data, generating reliable long event sequences from original incomplete ones. Our data synthesis method and its rationality and feasibility will be discussed in Chapter 5. In terms of algorithms, we can introduce more flexible point process models and learn the models in a discriminative way. Our discriminative learning algorithm can be explained as an implementation of neural networks, and Chapter 6 will explain our method in detail.

Innovative Applications of Point Processes. Point processes are now widely used in the field of data analysis and data mining. One might wonder whether we can extend the applications of point processes to other fields. In this thesis, we give a positive answer to this question and establish a point process-base visual attractiveness model. To the best of our knowledge, this is a pioneering work using point processes to solve problems of computer vision. Based on this work, we can select attractive clips from movies and generate trailers accordingly. The proposed attractiveness model and the whole trailer generation system will be introduced in Chapter 7

1.3 Literature Survey

1.3.1 Continuous-time Models

Sequential data modeling is a classical problem in the field of machine learning and signal processing. Many traditional models, such as Markov chain (MC) model [11], vector autoregressive (VAR) model [3, 12, 5], and hidden Markov model (HMM) [13, 14, 15], can be used to solve the problem when the sequential data is time series with fixed time lag. Recently, many efforts have been made to extend the models above from the discrete time domain to the continuous one. The continuous-time Markov chain (CTMC) is proposed in [16] to model the Markov chain in the continuous time domain, which can be viewed

as a special case of semi-Markov models [17]. Similarly, a hidden Markov model in the continuous time domain is proposed in [18].

Point processes are a kind of classic tools for modeling continuous-time event sequences [1]. Many different point processes have been proposed for various applications, e.g., Hawkes processes for social network modeling [19, 20, 21] and information system analysis [22, 23], and self-correcting processes for earthquake prediction [24, 25] and vision perception modeling [26]. An advantage of these point process models is considering the influence of all historical events on current one, which make these models outperform traditional low-order Markovian models. Recently, some work starts to apply point processes to analyze electronic health records for health information systems [7, 27].

1.3.2 Typical Point Processes

Many kinds of point processes have been proposed to capture diverse phenomena of interests, e.g., Poisson process, Hawkes process [28], and self-correcting process [24, 26].

- **Poisson Processes.** Poisson process is the simplest point process model, whose conditional intensity function is a constant (i.e., $\lambda_c = \mu_c$ for $c = 1, \dots, C$). This means that the current event is independent of the historical events. Unfortunately, most event sequences in practical situations do not obey such a strong assumption.
- **Hawkes Processes.** Hawkes process [28] provides us with an additive model to capture the triggering pattern of the event types quantitatively. Specifically, its conditional intensity have a particular form:

$$\lambda_c(t) = \underbrace{\mu_c}_{\text{exogenous}} + \underbrace{\sum_{c'=1}^C \int_0^t \phi_{cc'}(t, s) dN_{c'}(s)}_{\text{endogenous}}, \quad (1.5)$$

where μ_c is the exogenous base intensity independent of the history while $\int_0^t \phi_{cc'}(t, s) dN_{c'}(s)$ is the endogenous intensity capturing the influence of the his-

torical events on the type- c ones at time t [29]. Here, $\phi_{cc'}(t, s) \geq 0$ is called *impact function*. It quantifies the influence of the type- c' event at time s to the type- c event at time t . Hawkes processes provide us with a physically-meaningful model to capture the infectivity among various event types hidden in complicated event sequences, which have been widely used in social network analysis [30, 21], behavior analysis [19, 22], and financial analysis [31]. Existing work mainly uses predefined impact function with known parameters, e.g., the exponential functions in [32, 33, 34, 35, 23] and the power-law functions in [21]. For enhancing the flexibility, a nonparametric model of 1-D Hawkes process is first proposed in [36] based on ordinary differential equation (ODE), and this model is extended to multi-dimensional case in [30, 22]. Another nonparametric strategy is the contrast function-based estimation in [37, 38]. It minimizes the estimation error of conditional intensity function and leads to a Least-Squares (LS) problem [39]. Similarly, the work in [40] proposes a nonparametric estimation of Hawkes processes via solving the Wiener-Hopf equation. In [41, 42], the impact functions are decomposed into basis functions to avoid discretization. However, the methods in these references obey the assumption that the impact function is shift-invariant (i.e., $\phi_{cc'}(t, s) = \phi_{cc'}(t-s), t \geq s$), which limits their applications in practice. Recently, many attempts have been made to further enhance the flexibility of Hawkes processes, e.g., the time-dependent Hawkes process (TiDeH) in [43] and the neural Hawkes process in [44]. These methods take advantage of deep learning architecture and learn very flexible Hawkes processes from a large amount of data or features.

- **Self-Correcting Processes.** Different from Hawkes process, self-correcting process [24] takes advantage of a multiplicative model to represent its conditional in-

tensity function:

$$\begin{aligned}\lambda_c(t) &= \exp\left(\mu_c t - \sum_{c'=1}^C \phi_{cc'} N_{c'}(t)\right) \\ &= \underbrace{\exp(\mu_c t)}_{\text{exogenous}} \times \underbrace{\exp\left(-\sum_{c'=1}^C \phi_{cc'} N_{c'}(t)\right)}_{\text{endogenous}}.\end{aligned}\tag{1.6}$$

Here the exogenous part increases/decreases exponentially w.r.t. the time while the new coming event corrects the tendency. Compared with Hawkes process, the self-correcting process is more flexible because it does not require its parameters (i.e., $\{\mu_c, \phi_{cc'}\}$) to be non-negative.

1.3.3 Causality Analysis

Some efforts have been made to learn Granger causality of asynchronous event sequences based on point processes [12]. For general random processes, a kernel independence test is developed in [45]. Focusing on 1-D point processes with simple piecewise constant conditional intensity, a model for capturing temporal dependencies between event types is proposed in [46]. In [6, 47], the inherent grouping structure is considered when learning the Granger causality on networks from discrete transition process. The work in [48] comes up with a continuous-time diffusion network inference method based on parametric cascade generative process. In more general cases, a class of graphical models of marked point processes is proposed in [2] to capture the local independence over various marks. The work in [39] focuses on Hawkes processes, and connects the Granger causality graph of Hawkes process with its impact functions.

1.3.4 Clustering Analysis

Traditional clustering methods in [8, 9, 10] can only deal with time series, which are just synchronous (or aggregated) event data with discrete and fixed time-lagged variables. These methods rely on probabilistic mixture models [49], extracting features from sequen-

tial data and then learning clusters via the Gaussian mixture model (GMM) [50, 51, 52]. Recently, a mixture model of Markov chains is proposed in [53], which learns potential clusters from aggregated event data. To asynchronous event sequences, these clustering methods can be categorized into feature-based methods, which are based on learned or pre-defined features. A typical example is the work in [7]: a Gaussian process-base multi-task learning method is proposed to learn the patterns of the event sequences from predefined features. Similarly, the method in [22] also applies multi-task learning algorithm, which extracts the features of multiple Hawkes processes simultaneously and then learns the clusters via k-means. However, these methods cannot decide the number of clusters automatically. To solve this problem, Dirichlet process [54] and its variants [55] are used to model the prior distribution of clusters, and Bayesian inference algorithm is applied to learn models. Focusing on Hawkes processes, some feature-based mixture models are proposed [19, 56], which combine Hawkes processes with Dirichlet processes. However, these methods aim at modeling the clusters of the events hidden in the event sequences (i.e., sub-sequence clustering), rather than learning the clusters of the event sequences. To the best of our knowledge, the model-based clustering method has been rarely considered in the case of asynchronous event sequences.

1.3.5 Learning from Imperfect Observations

Learning point processes often suffers from imperfect observations, e.g., incomplete and noisy sequences. Many efforts have been made to solve this problem. In survival analysis, many techniques have been made to deal with truncated and censored data [57, 58, 59, 60]. Focusing on point processes, the global [61] or local [62] likelihood maximization methods are used to learn Poisson processes. To learn non-homogeneous Poisson processes, non-parametric approaches are proposed based on the pseudo maximum likelihood estimator (MLE) [63] or full MLE [64]. Theoretically, multiple imputation (MI) [65] is a general framework to stochastically impute incomplete or missing data from the current model. For

time series, bootstrap method [66, 67, 68, 69] and its variants [70, 71] have been used to improve learning results when observations are insufficient. Recently, the bootstrap methods have been introduced to the inference of point processes [72, 71, 73]. To learn Hawkes processes robustly, structural constraints, e.g., low-rank [22] and group-sparse regularizers [29], are introduced in the learning phase.

Besides incomplete and noisy observations, imbalanced data are also common in practice, which have negative influences on the learning of point processes. Many methods have been proposed to learn models from imbalanced data. Generally, these methods can be categorized into two classes. One kind of the methods is merging minor classes together and learning binary classifiers step-by-step [74, 75]. Another is weighting the samples to re-balance the data [76, 77, 78], where the samples in the minor classes have large weights while those in the major ones have small weights. In [79], the weights are added to unlabeled samples when training logistic regression, which can be viewed as the prior knowledge of model. Recently, the imbalanced data processing methods based on auxiliary samples are proposed. In [80], a semi-supervised dictionary learning algorithm is proposed to train a robust classifier from extremely few samples. Unlabeled samples are used as auxiliary samples in the training phase and added to minor classes adaptively. Focusing on the problem of data synthesis, auxiliary data are generated based on manifold assumptions in [81, 82].

1.4 Contributions and Organizations

To solve the four research problems above, throughout this dissertation, we mainly take the following three aspects into consideration:

Flexible Modeling: Based on traditional point process models like Hawkes processes and self-correcting processes, we propose a series of novel probabilistic models which have physically-meaningful explanations. In particular, we propose nonparametric Hawkes processes in Chapter 2 with the help of basis representation. In Chapter 5, we further extend the

model from time-invariant Hawkes processes to time-varying ones to model asynchronous event sequences on longtime scale. In the case that event sequences are generated via different Hawkes processes, we propose a multi-task multi-dimensional Hawkes process model in Chapter 3 and a Dirichlet mixture model of Hawkes processes in Chapter 4, respectively. Besides Hawkes processes, we also design a parametric self-correcting process model in Chapter 7 and extend it to a more flexible mutually-correcting process model in Chapter 6. These models are robust to model misspecification problem, which can be used to analyze various data sets.

Robust Learning: Given flexible models, we need robust algorithms to learn the models' parameters from asynchronous, complicated but imperfect event sequences. Based on maximum likelihood estimation (MLE), we propose a generative learning framework for our Hawkes process model and its variants in Chapters 2, 3, and 4. In our algorithmic framework, multiple techniques are applied, including the expectation-maximization (EM) algorithm, the alternating direction method of multipliers (ADMM), augmented Lagrangian method, and the numerical method of ordinary differential equations (ODE). Taking the prior knowledge of data into consideration, we impose various structural regularizers into our learning algorithms (e.g., sparse, low-rank, and group-lasso) and improve our learning results accordingly. Furthermore, we enhance the robustness of our learning algorithms in two directions. In Chapter 5, when it comes to short doubly-censored event sequences, we add a data synthesis method as the pre-processing of the training data. This method suppresses the risk of overfitting effectively. In Chapter 6, we connect the learning algorithm of point processes with the architecture of neural networks, learning point process models in a discriminative way and predicting the appearances of future events with high accuracy. These algorithms converge well. Compared with the state-of-art algorithms, our learning algorithms have lower computational complexity and obtain better experimental results on both synthetic and real-world data sets.

Broad Applications: We apply our models and learning algorithms to diverse applica-

tions. Specifically, the applications mentioned in this thesis include 1) IPTV user behavior analysis and user clustering (Chapters 2, 3, and 4); 2) ICU patient flow modeling, clustering, and prediction (Chapters 4 and 6); 3) constructing dynamical disease networks based on patients' admission records (Chapter 5); 4) constructing dynamical infectivity graph of IT companies (Chapter 5); 5) modeling visual attractiveness of video clips (Chapter 7). These applications cover many interesting problems in the field of social behavior analysis, smart health, and computer vision, which demonstrate the usefulness of our work.

In summary, we present our contributions in these three aspects and the organization of this dissertation in Table 1.1.

Table 1.1: Contributions and Organizations of This Dissertation

| RESEARCH TASK | CHAPTER | MODEL | ALGORITHM | APPLICATION |
|------------------------|-----------|---------------------------------------|----------------|-------------------------|
| CAUSALITY ANALYSIS | CHAPTER 2 | TIME-INVARIANT HAWKES PROCESSES | GENERATIVE | IPTV SYSTEM ANALYSIS |
| CLUSTERING ANALYSIS | CHAPTER 3 | MULTI-TASK HAWKES PROCESSES | GENERATIVE | IPTV USER CLUSTERING |
| CLUSTERING ANALYSIS | CHAPTER 4 | DIRICHLET MIXTURE OF HAWKES PROCESSES | GENERATIVE | ICU PATIENT CLUSTERING |
| ROBUST LEARNING | CHAPTER 5 | TIME-VARYING HAWKES PROCESSES | GENERATIVE | DISEASE NETWORKS |
| ROBUST LEARNING | CHAPTER 6 | MUTUALLY-CORRECTING PROCESSES | DISCRIMINATIVE | PATIENT FLOW PREDICTION |
| INNOVATIVE APPLICATION | CHAPTER 7 | SELF-CORRECTING PROCESSES | GENERATIVE | TRAILER GENERATION |

CHAPTER 2

LEARNING GRANGER CAUSALITY BASED ON HAWKES PROCESSES

2.1 Introduction

In many practical situations, we need to deal with a large amount of asynchronous event sequences with multiple event types, e.g., viewing records in an IPTV system (when and which TV programs are viewed) and patient records in hospitals (when and what diagnoses and treatments are given), which can be modeled via multi-dimensional point processes. A significant task is to learn the so-called Granger causality among different event types from the event sequences. From the viewpoint of graphical models, it means to construct a directed graph called Granger causality graph (or local independence graph) [2] based on a point process model, in which the nodes are the dimensions of the point process (i.e., the event types appearing in the sequences) and the arrow connecting two nodes indicates the Granger causation from the source node to the destination one. Learning Granger causality from asynchronous event sequences is meaningful for many practical applications. Take our previous two examples: the Granger causality among IPTV programs reflects users' viewing preferences and patterns, which is important for personalized program recommendation and IPTV system simulation; the Granger causality among diseases helps us to construct a disease network, which is beneficial to predict potential diseases for patients and leads to more effective treatments.

Unfortunately, learning Granger causality is very challenging for asynchronous event sequences. As aforementioned, existing work mainly focuses on learning Granger causality from time series [3, 4, 6], where the Granger causality is captured via the vector autoregressive (VAR) model [5]. For asynchronous event sequences, whose events are non-fixed time-lagged observations in the continuous time domain, it is hard to find a univer-

sal and tractable representation of events to describe Granger causality. Although some feature-based methods [7] are proposed to learn Granger causality via feature selection. These methods are sensitive to the specific feature construction method used, resulting in dubious Granger causality.

To make concrete progress, we focus on a special class of point processes called Hawkes processes and their Granger causality. Hawkes processes are capable of describing the triggering pattern of event types, whose applications include bioinformatics [37], social network analysis [21], and financial analysis [31], etc. Technically, based on the graphical model of point process [2], the Granger causality of Hawkes process can be captured by its impact functions. Inspired by this fact, we propose a nonparametric model of Hawkes processes, where the impact functions are represented by a series of basis functions, and we discover the Granger causality via the group sparsity of the impact functions' coefficients. Based on the explicit representation of Granger causality, we propose a novel learning algorithm combining the maximum likelihood estimator with the sparse-group-lasso (SGL) regularizer on impact functions. The pairwise similarity between various impact functions is imposed when the clustering structure of the event types is available. These structural constraints enhance the robustness of our method. The learning algorithm applies the EM-based strategy [36, 34] and obtains close-form solutions to update model's parameters iteratively.

Compared with existing learning methods of Hawkes processes [30, 39], our model avoids discretized representation of impact functions and conditional intensity, and considers the induced structures across impact functions. These improvements reduce the complexity of algorithm and help us to learn the Granger causality graph with high accuracy. We investigate the robustness of our method and test our method on both synthetic and real-world data. Experimental results show that our method can indeed reveal the Granger causality for asynchronous event sequences and obtain superior learning performance compared with other competitors.

2.2 Granger Causality Based on Point Processes

Given a set of event sequences generated via a C -dimensional point processes, where $1, \dots, C \in \mathcal{C}$ are event types, we are interested in identifying, if possible, a subset of the event types $\mathcal{V} \subset \mathcal{C}$ for the type- c event, such that $\lambda_c(t)$ only depends on historical events of types in \mathcal{V} , denoted as $\mathcal{H}_t^\mathcal{V}$, and not those of the remaining types, denoted as $\mathcal{H}_t^{\mathcal{C} \setminus \mathcal{V}}$. In the viewpoint of graphical model, it is about local independence over the dimensions of the point process — the occurrence of historical events in \mathcal{V} influences the probability of occurrence of type- c events at present and future while the occurrence of historical events in $\mathcal{C} \setminus \mathcal{V}$ does not. In order to proceed formally we introduce some notations. For a subset $\mathcal{V} \subset \mathcal{C}$, let $N_\mathcal{V} = \{N_c(t) | c \in \mathcal{V}\}$. The filtration $\mathfrak{F}_t^\mathcal{V}$ is defined as $\sigma\{N_c(s) | s \leq t, c \in \mathcal{V}\}$, i.e., the smallest σ -algebra generated by the random processes. In particular, \mathfrak{F}_t^c is the internal filtration of the counting process $N_c(t)$ while \mathfrak{F}_t^{-c} is the filtration for the subset $\mathcal{C} \setminus \{c\}$.

Definition 2.2.1. [2]. *The counting process N_c is locally independent of $N_{c'}$ given $N_{\mathcal{C} \setminus \{c, c'\}}$ if the intensity function $\lambda_c(t)$ is measurable w.r.t. $\mathfrak{F}_t^{-c'}$ for all $t \in [T_b, T_e]$. Otherwise N_c is locally dependent on $N_{c'}$.*

Intuitively, the above definition says that $\{N_{c'}(s) | s < t\}$ does not influence $\lambda_c(t)$, given $\{N_l(s) | s < t, l \neq c'\}$. In [39], the notion of Granger non-causality is used, and the above definition is equivalent to saying that type- c' event does not Granger-cause type- c event w.r.t. \mathfrak{F}_t^c . Otherwise, we say type- c' event Granger-causes type- c event w.r.t. \mathfrak{F}_t^c . With this definition, we can construct the so-called *Granger causality graph* $G = (\mathcal{C}, \mathcal{E})$ with the event types \mathcal{C} (i.e., the dimensions of the point process) as the nodes and the directed edges indicating the causation (i.e., $c' \rightarrow c \in \mathcal{E}$ if type- c' event Granger-causes type- c one).

2.3 Proposed Model

Although learning Granger causality is a difficult problem given a general multi-dimensional point process model, we can design an efficient method for learning the Granger causality based on Hawkes process. In particular, we propose a model of Hawkes process representing impact functions via a series of basis functions. An efficient learning algorithm combining the MLE with the sparse-group-lasso is applied and analyzed in details. Compared with existing learning algorithms, our algorithm is based on convex optimization and has lower complexity, which learns Granger causality robustly.

2.3.1 Granger Causality Based on Time-invariant Hawkes Process

In this chapter, we consider a simple Hawkes process model with time-invariant impact functions. Specifically, its conditional intensity functions have a particular form:

$$\begin{aligned}\lambda_c(t) &= \mu_c + \sum_{c'=1}^C \int_{T_b}^t \phi_{cc'}(t-s) dN_{c'}(s) \\ &= \mu_c + \sum_{c'=1}^C \int_{T_b}^t \phi_{cc'}(s) dN_{c'}(t-s),\end{aligned}\tag{2.1}$$

where μ_c is the exogenous base intensity independent of the history while $\sum_{c'=1}^C \int_{T_b}^t \phi_{cc'}(t-s) dN_{c'}(s)$ is the endogenous intensity capturing the peer influence [32]. The impact function (i.e., $\phi_{cc'}(t, s) = \phi_{cc'}(t-s) \geq 0$, $T_b \leq s < t \leq T_e$) is assumed to be time-invariant. Focusing on the time-invariant Hawkes process model, the work in [39] reveals the relationship between the Hawkes process's impact functions and its Granger causality graph as follows,

Theorem 2.3.1. [39]. *Assume a Hawkes process with conditional intensity function defined in (2.1) and a Granger causality graph $G(\mathcal{C}, \mathcal{E})$. If the condition $dN_{c'}(t-s) > 0$ for $T_b \leq s < t \leq T_e$ holds, then, $c' \rightarrow c \notin \mathcal{E}$ if and only if $\phi_{cc'}(t) = 0$ for $t \in [0, \infty]$.*

In practice, Theorem 4.3.1 can be easily specified in the time interval $[T_b, T_e]$. It pro-

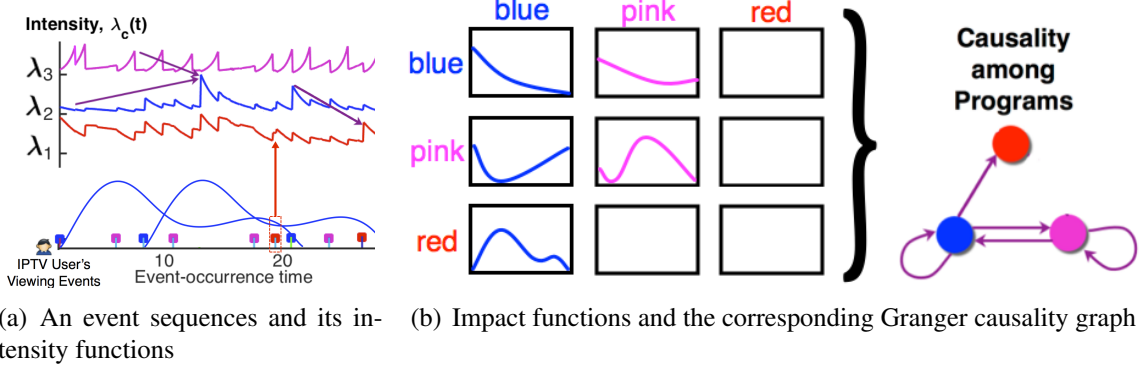


Figure 2.1: In the left subfigure, an event sequence generated via a 3-dimensional Hawkes process is given, where the event with red type is only Granger caused via historical events with blue type. In the right subfigure, the group sparsity of learned impact functions indicate the structure of the Granger causality graph.

vides an explicit representation of the Granger causality — learning whether type- c' event Granger-causes type- c event or not is equivalent to detecting whether the impact function $\phi_{cc'}(t)$ is all-zero or not. In other words, we can take each impact function as a group, and the group sparsity of the impact functions indicates the Granger causality graph over the dimensions of Hawkes process. Fig. 2.1 illustrates the principle of our learning strategy. Therefore, for multi-dimensional Hawkes process, we can learn its Granger causality via learning its impact functions, which requires tractable and flexible representations of the functions.

2.3.2 Learning Task

When we parameterize $\phi_{cc'}(t) = a_{cc'}\kappa(t)$ as [34] does, where $\kappa(t)$ models the time-decay of event's influence and $a_{cc'} \geq 0$ captures the influence of type- c' events on type- c ones, the *infectivity matrix* $\mathbf{A} = [a_{cc'}]$ is the adjacency matrix of the corresponding Granger causality graph. Although such a parametric model simplifies the representation of impact function and reduces the complexity of the model, this achievement comes with the cost of inflexibility of the model — the model estimation will be poor if the data does not conform to the assumptions of the model. To address this problem, we propose a nonparametric model

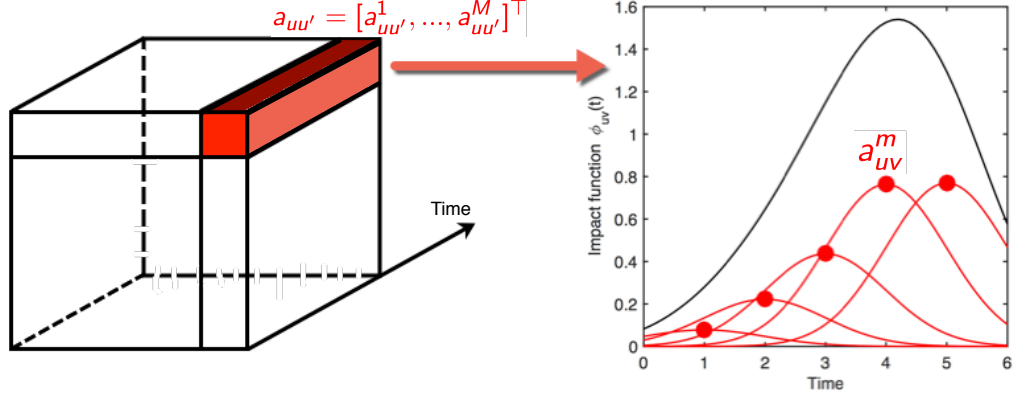


Figure 2.2: Each vector in the tensor of parameters represents the coefficients of an impact function.

of Hawkes processes, representing the impact function in (2.1) via a linear combination of basis functions as

$$\phi_{cc'}(t) = \sum_{m=1}^M a_{cc'm} \kappa_m(t). \quad (2.2)$$

Here $\kappa_m(t)$ is the m -th basis function and $a_{cc'm}$ is the coefficient corresponding to $\kappa_m(t)$. Therefore, the parameters of the Hawkes process is $\Theta = \{\mathbf{A} = [a_{cc'm}] \in \mathbb{R}^{C \times C \times M}, \boldsymbol{\mu} = [\mu_c] \in \mathbb{R}^C\}$. Fig. 2.2 illustrates the tensor of parameters and the basis representation of an impact function.

Suppose we have a set of event sequences $\mathcal{S} = \{\mathbf{s}_n\}_{n=1}^N$. $\mathbf{s}_n = \{(t_i^n, c_i^n)\}_{i=1}^{I_n}$, where $t_i^n \in [T_b^n, T_e^n]$ is the time stamp of the i -th event of \mathbf{s}_n and $c_i^n \in \{1, \dots, C\}$ is the event type.

The log-likelihood function can be expressed as:

$$\begin{aligned} \log \mathcal{L}(\mathcal{S}; \Theta) &= \sum_{n=1}^N \left\{ \sum_{i=1}^{I_n} \log \lambda_{c_i^n}(t_i^n) - \sum_{c=1}^C \int_{T_b^n}^{T_e^n} \lambda_c(s) ds \right\} \\ &= \sum_{n=1}^N \left\{ \sum_{i=1}^{I_n} \log \left(\mu_{c_i^n} + \sum_{j=1}^{i-1} \sum_{m=1}^M a_{c_i^n c_j^n m} \kappa_m(\tau_{ij}^n) \right) \right. \\ &\quad \left. - \sum_{c=1}^C \left((T_e^n - T_b^n) \mu_c + \sum_{i=1}^{I_n} \sum_{m=1}^M a_{cc_i^n m} K_m(T_e^n - t_i^n) \right) \right\}, \end{aligned} \quad (2.3)$$

where $\tau_{ij}^n = t_i^n - t_j^n$, $K_m(t) = \int_0^t \kappa_m(s)ds$. To construct Granger causality accurately and robustly, we consider the following three regularizers:

Local Independence. According to Theorem 4.3.1, the type- c' event has no influence on the type- c event (i.e., directed edge $c' \rightarrow c \notin \mathcal{E}$) if and only if $\phi_{cc'}(t) = 0$ for all $t \in \mathbb{R}$, which requires $a_{cc'}^m = 0$ for all m . Therefore, we use group-lasso [83, 84, 47] to regularize the coefficients of impact functions, denoted as $\|\mathbf{A}\|_{1,2} = \sum_{c,c'} \|a_{cc'}\|_2$, where $a_{cc'} = [a_{cc'1}, \dots, a_{cc'M}]^\top$. This means that along the time dimension the coefficients' tensor \mathbf{A} should obey the constraint of group sparsity.

Temporal Sparsity. A necessary condition for the stationarity of Hawkes process is $\int_0^\infty \phi_{cc'}(s)ds < \infty$, which means $\lim_{t \rightarrow \infty} \phi_{cc'}(t) \rightarrow 0$. Therefore, we add sparsity constraints to the coefficients of impact functions, denoted as $\|\mathbf{A}\|_1 = \sum_{c,c',m} |a_{cc'm}|$.

Pairwise Similarity. Event types of Hawkes process may exhibit clustering structure. For example, if c and c' are similar event types, their influences on other event types should be similar (i.e., $\phi_{\cdot c}(t) \approx \phi_{\cdot c'}(t)$) and the influences of other event types on them should be similar as well (i.e., $\phi_{c \cdot}(t) \approx \phi_{c' \cdot}(t)$). When the clustering structure is available, we can add the pairwise similarity to the coefficients of the impact functions as follows:

$$E(\mathbf{A}) = \sum_{c=1}^C \sum_{c' \in \mathcal{C}_c} \|a_{c \cdot} - a_{c' \cdot}\|_F^2 + \|a_{\cdot c'} - a_{\cdot c}\|_F^2.$$

\mathcal{C}_c contains the event types within the cluster that the type- c event resides. $a_{c \cdot} \in \mathbb{R}^{C \times M}$ is the slice of \mathbf{A} with row index c , and $a_{\cdot c} \in \mathbb{R}^{C \times M}$ is the slice with column index c . In summary, the learning problem of the Hawkes process is

$$\min_{\Theta \geq 0} -\log \mathcal{L}(\mathcal{S}; \Theta) + \alpha_S \|\mathbf{A}\|_1 + \alpha_G \|\mathbf{A}\|_{1,2} + \alpha_P E(\mathbf{A}). \quad (2.4)$$

Here the parameters $\{\alpha_S, \alpha_G, \alpha_P\}$ control the influences of the regularizers. The non-negative constraint guarantees the model being physically-meaningful.

2.4 Learning Algorithm

2.4.1 An EM-based Algorithm

Following [36, 30], we propose an EM-based learning algorithm to solve the optimization problem (2.4) iteratively.

Step 1: Update \mathbf{A} and $\boldsymbol{\mu}$. Specifically, given current parameters $\boldsymbol{\Theta}^{(k)}$, we first construct a tight bound of log-likelihood function appeared in (5.9) as follows:

$$\begin{aligned} \mathcal{Q}(\boldsymbol{\Theta}; \boldsymbol{\Theta}^{(k)}) = & \sum_{n=1}^N \left\{ - \sum_{c=1}^C \left((T_e^n - T_b^n) \mu_c + \sum_{i=1}^{I_n} \sum_{m=1}^M a_{cc_i^n m} K_m(T_e^n - t_i^n) \right) \right. \\ & \left. + \sum_{i=1}^{I_n} \left(p_{ii} \log \frac{\mu_{c_i^n}}{p_{ij}} + \sum_{j=1}^{i-1} \sum_{m=1}^M p_{ijm} \log \frac{a_{c_i^n c_j^n m} \kappa_m(\tau_{ij}^n)}{p_{ijm}} \right) \right\}, \end{aligned} \quad (2.5)$$

where

$$p_{ii} = \frac{\mu_{c_i^n}^{(k)}}{\lambda_{c_i^n}^{(k)}(t_i^n)}, \quad p_{ijm} = \frac{a_{c_i^n c_j^n m}^{(k)} \kappa_m(\tau_{ij}^n)}{\lambda_{c_i^n}^{(k)}(t_i^n)}.$$

$\lambda_c^{(k)}(t)$ is the conditional intensity function computed with current parameters. The rationality of (B.1) is based on Jensen's inequality. Specifically, we have following inequality for all n and i :

$$\log \left(\mu_{c_i^n} + \sum_{m=1}^M \sum_{j=1}^{i-1} a_{c_i^n c_j^n m} \kappa_m(\tau_{ij}^n) \right) \geq p_{ii} \log \left(\frac{\mu_{c_i^n}}{p_{ii}} \right) + \sum_{m=1}^M \sum_{j=1}^{i-1} p_{ijm} \log \left(\frac{a_{c_i^n c_j^n m} \kappa_m(\tau_{ij}^n)}{p_{ijm}} \right).$$

The equation holds if and only if $\mu_c = \mu_c^{(k)}$ and $a_{cc'm} = a_{cc'm}^{(k)}$. Therefore, we have $\mathcal{Q}(\boldsymbol{\Theta}; \boldsymbol{\Theta}^{(k)}) \leq \log \mathcal{L}(\mathcal{S}; \boldsymbol{\Theta})$ and $\mathcal{Q}(\boldsymbol{\Theta}^{(k)}; \boldsymbol{\Theta}^{(k)}) = \log \mathcal{L}(\mathcal{S}; \boldsymbol{\Theta}^{(k)})$. Additionally, when the constraint of the pairwise similarity is used, we rewrite $E(\mathbf{A})$ given the current parameters as

$$E(\boldsymbol{\Theta}; \boldsymbol{\Theta}^{(k)}) = \sum_{c=1}^C \sum_{c' \in \mathcal{C}_c} \|a_{c\cdot} - a_{c'}^{(k)}\|_F^2 + \|a_{\cdot c'} - a_{\cdot c}^{(k)}\|_F^2.$$

Replacing $\mathcal{L}(\mathcal{S}; \Theta)$ and $E(\mathbf{A})$ with $\mathcal{Q}(\Theta; \Theta^{(k)})$ and $E(\Theta; \Theta^{(k)})$, respectively, we decouple the parameters and obtain the surrogate objective function

$$F = -\mathcal{Q}(\Theta; \Theta^{(k)}) + \alpha_S \|\mathbf{A}\|_1 + \alpha_G \|\mathbf{A}\|_{1,2} + \alpha_P E(\Theta; \Theta^{(k)}).$$

Then, we update each individual parameter via solving $\frac{\partial F}{\partial \Theta} = \mathbf{0}$, and obtain the following closed-form updates:

$$\mu_c^{(k+1)} = \frac{\sum_{n=1}^N \sum_{c_i^n=c} p_{ii}}{\sum_{n=1}^N (T_e^n - T_b^n)}, \quad a_{cc'm}^{(k+1)} = \frac{-B + \sqrt{B^2 - 4AC}}{2A}, \quad (2.6)$$

where

$$A = \frac{\alpha_G}{\|a_{cc'}^{(k)}\|_2} + 2(|\mathcal{C}_c| + |\mathcal{C}_{c'}|)\alpha'_P, \quad \alpha'_P = \begin{cases} \alpha_P, & c' \in \mathcal{C}_c \\ 0, & \text{others} \end{cases}$$

$$B = \sum_{n=1}^N \sum_{c_i^n=c'} K_m(T_e^n - t_i^n) + \alpha_S - 2\alpha'_P \left(\sum_{v \in \mathcal{C}_c} a_{vc'm}^{(k)} + \sum_{v' \in \mathcal{C}_{c'}} a_{cv'm}^{(k)} \right),$$

$$C = - \sum_{n=1}^N \sum_{c_i^n=c} \sum_{c_j^n=c'} p_{ijm}.$$

Step 2: Impose Group Sparsity on \mathbf{A} . Furthermore, in order to solve the sparse-group-lasso (SGL), we apply the soft-thresholding method in [84] to shrink the updated parameters. Specifically, for each pair $\{c, c'\}$ we choose the group $a_{cc'}^{(k+1)} = [a_{cc'1}^{(k+1)}, \dots, a_{cc'M}^{(k+1)}]^\top$ to minimize and fix other parameters. Given current estimate $a_{cc'}^{(k+1)}$, we majorize $\mathcal{Q}(\Theta; \Theta^{(k+1)})$ as

$$\mathcal{Q} \leq \mathcal{Q}|_{a_{cc'}^{(k+1)}} + (a_{cc'} - a_{cc'}^{(k+1)}) (\nabla_{a_{cc'}} \mathcal{Q})|_{a_{cc'}^{(k+1)}} + \frac{1}{2\eta} \|a_{cc'} - a_{cc'}^{(k+1)}\|_2^2, \quad (2.7)$$

where $\nabla_x f$ is the subgradient of function f w.r.t. variable x , and $f|_{x_0}$ is the value of function f at x_0 . Introducing (2.7) to the surrogate objective function, we rewrite the

optimization problem as

$$\begin{aligned} \min_{a_{cc'} \geq 0} \quad & \mathcal{Q}|_{a_{cc'}^{(k+1)}} + (a_{cc'} - a_{cc'}^{(k+1)})(\nabla_{a_{cc'}} \mathcal{Q})|_{a_{cc'}^{(k+1)}} \\ & + \frac{1}{2\eta} \|a_{cc'} - a_{cc'}^{(k+1)}\|_2^2 + \alpha_S \|a_{cc'}\|_1 + \alpha_G \|a_{cc'}\|_2. \end{aligned} \quad (2.8)$$

Because both $\mathcal{Q}|_{a_{cc'}^{(k+1)}}$ and $(\nabla_{a_{cc'}} \mathcal{Q})|_{a_{cc'}^{(k+1)}}$ are known, we add $\frac{\eta}{2} \|(\nabla_{a_{cc'}} \mathcal{Q})|_{a_{cc'}^{(k+1)}}\|_2^2$ to the objective function of (2.8) and reduce $\mathcal{Q}|_{a_{cc'}^{(k+1)}}$ from it, and obtain an equivalent optimization problem:

$$\min_{a_{cc'} \geq 0} \quad \frac{1}{2\eta} \|a_{cc'} - (a_{cc'}^{(k+1)} - \eta(\nabla_{a_{cc'}} \mathcal{Q})|_{a_{cc'}^{(k+1)}})\|_2^2 + \alpha_S \|a_{cc'}\|_1 + \alpha_G \|a_{cc'}\|_2. \quad (2.9)$$

The objective function in (2.9) is convex, so the optimal solution is characterized by the subgradient equations:

$$a_{cc'}^{(k+1)} - \eta(\nabla_{a_{cc'}} \mathcal{Q})|_{a_{cc'}^{(k+1)}} - a_{cc'} = \eta\alpha_S \gamma + \eta\alpha_G \beta. \quad (2.10)$$

$\gamma = [\gamma_1, \dots, \gamma_M]^\top$, where $\gamma_m = 1$ if $a_{cc'm} > 0$, and in $[0, 1]$ otherwise. $\beta = \frac{a_{cc'}}{\|a_{cc'}\|_2}$ if $a_{cc'} \neq 0$, and in the set $\{x | \|x\|_2 \leq 1\}$ otherwise.

As a result, we update $a_{cc'}^{(k+1)}$ to all-zero if the following condition is holds:

$$\|S_{\eta\alpha_S}(a_{cc'}^{(k+1)} - \eta(\nabla_{a_{cc'}} \mathcal{Q})|_{a_{cc'}^{(k+1)}})\|_2 \leq \eta\alpha_G, \quad (2.11)$$

where $S_\alpha(z) = \text{sign}(z)(|z| - \alpha)_+$ achieves soft-thresholding for each element of input.

Otherwise, $a_{cc'}$ satisfies

$$\left(1 + \frac{\eta\alpha_G}{\|a_{cc'}\|_2}\right) a_{cc'} = S_{\eta\alpha_S}(a_{cc'}^{(k+1)} - \eta(\nabla_{a_{cc'}} \mathcal{Q})|_{a_{cc'}^{(k+1)}}), \quad (2.12)$$

Taking the norm on both sides, $\|a_{cc'}\|_2$ can be replaced by

$$(\|S_{\eta\alpha_S}(a_{cc'}^{(k)} - \eta(\nabla_{a_{cc'}}\mathcal{Q})|_{a_{cc'}^{(k)}})\|_2 - t\eta\alpha_G)_+. \quad (2.13)$$

Replacing the $\|a_{cc'}\|_2$ in (2.12) with (2.13), we obtain the generalized gradient step:

$$\begin{aligned} a_{cc'}^{(k+1)} = & \left(1 - \frac{\eta\alpha_G}{\|S_{\eta\alpha_S}(a_{cc'}^{(k+1)} - \eta(\nabla_{a_{cc'}}\mathcal{Q})|_{a_{cc'}^{(k+1)}})\|_2} \right)_+ \\ & \times S_{\eta\alpha_S}(a_{cc'}^{(k+1)} - \eta(\nabla_{a_{cc'}}\mathcal{Q})|_{a_{cc'}^{(k+1)}}) \end{aligned} \quad (2.14)$$

In summary, Algorithm 1 gives the scheme of our MLE-based algorithm with sparse-group-lasso and pairwise similarity constraints, which is called MLE-SGLP for short.

Algorithm 1 Learning Time-invariant Hawkes Processes (MLE-SGLP)

- 1: **Input:** Event sequences $\mathcal{S} = \{\mathbf{s}_n\}_{n=1}^N$, parameters α_S , α_G , (optional) clustering structure and α_P .
 - 2: **Output:** Parameters of model, $\boldsymbol{\mu}$ and \mathbf{A} .
 - 3: Initialize $\boldsymbol{\mu} = [\mu_c]$ and $\mathbf{A} = [a_{cc'm}]$ randomly.
 - 4: **repeat**
 - 5: **repeat**
 - 6: Update $\boldsymbol{\mu}$ and \mathbf{A} via (2.6).
 - 7: **until** convergence
 - 8: **for** $c, c' = 1 : C$
 - 9: **if** (2.11) holds, $a_{cc'} = 0$; **else**, update $a_{cc'}$ via (2.14).
 - 10: **until** convergence
-

2.4.2 Adaptive Selection of Basis Functions

Although the nonparametric models in [42, 30] represent impact functions as we do via a set of basis functions, they do not provide a guidance that how to select basis functions. A contribution of our work is proposing a method of selecting basis functions founded on sampling theory [85]. Specifically, we focus on the impact functions satisfying following assumptions.

Assumption 2.4.1. (i) $\phi(t) \geq 0$, and $\int_0^\infty \phi(t)dt < \infty$. (ii) For arbitrary $\epsilon > 0$, there always exists a ω_0 , such that $\int_{\omega_0}^\infty |\hat{\phi}(\omega)|d\omega \leq \epsilon$. $\hat{\phi}(\omega)$ is the Fourier transform of $\phi(t)$.

The assumption (i) guarantees the existence of $\hat{\phi}(\omega)$, while the assumption (ii) means that we can find a function with a bandlimit, denoted as $\frac{\omega_0}{2\pi}$, to approximate the target impact function with bounded residual.

Based on these two assumptions, the representation of impact function in (2.2) can be explained as a sampling process. The $\{a_{cc'm}\}_{m=1}^M$ can be viewed as the discretized samples of $\phi_{cc'}(t)$ in $[T_b, T_e]$ and $\kappa_m(t) = \kappa_\omega(t, t_m)$ is sampling function (i.e., sinc or Gaussian function¹) corresponding to a low-pass filter with cut-off frequency ω . t_m is the sampling location corresponding to $a_{cc'm}$ and the sampling rate is $\frac{\omega}{\pi}$. The Nyquist-Shannon theorem requires us to have $\omega = \omega_0$, at least, such that the sampling rate is high enough (i.e., $\frac{\omega_0}{\pi}$, twice bandlimit) to approximate the impact function. Accordingly, the number of samples is $M = \lceil \frac{T\omega_0}{\pi} \rceil$, where $\lceil x \rceil$ returns the smallest integer larger than or equal to x .

Based on the above argument, the core of selecting basis functions is estimating ω_0 . It is hard because we cannot observe impact functions directly. Fortunately, based on (2.1) we know that the bandlimits of the impact functions cannot be larger than that of conditional intensity functions $\lambda(t) = \sum_{c=1}^C \lambda_c(t)$. When sufficient training sequences $\mathcal{S} = \{\mathbf{s}_n\}_{n=1}^N$ are available, we can estimate $\lambda(t)$ via a Gaussian-based kernel density estimator:

$$\lambda(t) = \sum_{n=1}^N \sum_{i=1}^{I_n} G_h(t - t_i^n). \quad (2.15)$$

Here $G_h(\cdot)$ is a Gaussian kernel with the bandlimit h . Applying Silverman's rule of thumb [86], we set optimal $h = (\frac{4\hat{\sigma}^5}{3\sum_n I_n})^{0.2}$, where $\hat{\sigma}$ is the standard deviation of time stamps $\{t_i^n\}$. Therefore, given the upper bound of residual ϵ , we can estimate ω_0 from the Fourier transformation of $\lambda(t)$, which actually does not require us to compute $\lambda(t)$ via (2.15) directly. In summary, we propose Algorithm 2 to select basis functions and more

¹For Gaussian filter $\kappa_\omega(t, t_m) = \exp(-(t - t_m)^2/(2\sigma^2))$, its bandlimit is defined as $\omega = \sigma^{-1}$.

detailed analysis is given in Appendix A.

Algorithm 2 The selection of basis functions

- 1: **Input:** $\mathcal{S} = \{s_n\}_{n=1}^N$, residual's upper bound ϵ .
 - 2: **Output:** Basis functions $\{\kappa_{\omega_0}(t, t_m)\}_{m=1}^M$.
 - 3: Compute $\left(\sum_{n=1}^N I_n \sqrt{2\pi h^2}\right) e^{-\frac{\omega^2 h^2}{2}}$ to bound $|\hat{\lambda}(\omega)|$.
 - 4: Find the smallest ω_0 satisfying $\int_{\omega_0}^{\infty} |\hat{\lambda}(\omega)| d\omega \leq \epsilon$.
 - 5: The proposed basis functions $\{\kappa_{\omega_0}(t, t_m)\}_{m=1}^M$ are selected, where ω_0 is the cut-off frequency of basis function and $t_m = \frac{(m-1)T}{M}$, $M = \lceil \frac{T\omega_0}{\pi} \rceil$.
-

2.4.3 Properties of The Proposed Method

Compared with existing state-of-art methods, e.g., the learning algorithm of the Hawkes process with single exponential kernel (ExpKernel) in [34], the ODE-based algorithm in [30], and the Least-Squares (LS) algorithm in [39], our MLE-SGLP algorithm has several advantages.

Computational complexity: Given a training sequence with I events generated via a C -dimensional Hawkes process, the ODE-based algorithm in [30] represents impact functions by M basis functions, where each basis function is discretized to L points. The basis functions are updated iteratively via solving M Euler-Lagrange equations. The complexity of the ODE-based algorithm per iteration is $\mathcal{O}(MI^3C^2 + ML(IC + I^2))$. The LS algorithm in [39] directly discretizes the timeline into L small intervals. In such a situation, impact functions are discretized to L points. The computational complexity of the algorithm is $\mathcal{O}(IC^3L^3)$. In contrast, our algorithm is based on known basis functions and does not estimate impact function via discretized points. The computational complexity of our algorithm per iteration is $\mathcal{O}(MI^3C^2)$. To get accurate estimation, the ODE-based algorithm sampling basis functions densely, and the LS algorithm needs to ensure that there is at most one event in each interval. In other words, both two competitors require $L \gg I$. Therefore, the computational complexity of the LS algorithm is the highest among the three, and the complexity of our algorithm is at least comparable to that of the ODE-based algo-

rithm. Although the computational complexity of the ExpKernel-based algorithm in [34] is $\mathcal{O}(I^3U^2)$, which is lower than ours, using single exponential kernel would lead to serious model misspecification problem.

Convexity: The ExpKernel algorithm, the LS algorithm, and our algorithm are convex and can achieve global optima. The ODE-based algorithm, however, learns basis functions and their weights alternatively. It is not convex and is prone to a local optima.

Inference of Granger causality: Neither the ODE-based algorithm nor the LS algorithm considers to infer the Granger causality graph when learning model. Without suitable regularizers on impact functions, the impact functions learned by these two algorithms are non-zero generally, which cannot indicate the Granger causality graph exactly. What is worse, the LS algorithm even may obtain physically-meaningless impact functions with negative values. Although the ExpKernel algorithm applies sparse regularizer to parameters of model, the model misspecification problem caused via the algorithm is serious. To the best of our knowledge, our algorithm is the first attempt to solving this problem via combining the MLE of the Hawkes process with the sparse-group-lasso regularizer, which learns the Granger causality graph robustly.

In summary, the comparison for various methods is shown in Table 2.1.

Table 2.1: Comparison for various methods

| METHOD | COMPLEXITY PER ITERATION | CONVEXITY | CAUSALITY | CAPACITY |
|----------------|---------------------------------------|-----------|-----------|----------|
| MLE-SGLP | $\mathcal{O}(MI^3C^2)$ | YES | YES | HIGH |
| LS [39] | $\mathcal{O}(IC^3L^3)$ | YES | NO | HIGH |
| ODE [30] | $\mathcal{O}(MI^3C^2 + ML(IC + I^2))$ | NO | NO | HIGH |
| EXPKERNEL [34] | $\mathcal{O}(I^3C^2)$ | YES | NO | LOW |

2.5 Experiments

To demonstrate the feasibility and the efficiency of our algorithm (MLE-SGLP), we compare it with the state-of-art methods — the ODE-based method in [30] and the Least-

Squares (LS) method in [39] — on both synthetic and real-world data. We also investigate the influences of regularizers via comparing our algorithm with its variants, including the pure MLE without any regularizer (MLE), the MLE with group-lasso (MLE-GL), and the MLE with sparse regularizer (MLE-S). To evaluate algorithms comprehensively, given estimate $\tilde{\Theta} = \{\tilde{\boldsymbol{\mu}}, \tilde{\mathbf{A}}\}$, we apply the following measurements:

- 1) The log-likelihood of testing data, denoted as *Loglike*;
- 2) the relative error of $\boldsymbol{\mu}$, $e_{\mu} = \frac{\|\tilde{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_2}{\|\boldsymbol{\mu}\|_2}$;
- 3) the relative error of $\Phi(t) = [\phi_{cc'}(t)]$, $e_{\phi} = \frac{1}{C^2} \sum_{c,c'} \frac{\int_0^T |\tilde{\phi}_{cc'}(t) - \phi_{cc'}(t)| dt}{\int_0^T \phi_{cc'}(t) dt}$;
- 4) *Sparsity of impact function* — the Granger causality graph is indicated via all-zero impact functions.

2.5.1 Synthetic Data

We generate two synthetic data sets using sine-like impact functions and piecewise constant impact function respectively. Each of them contains 500 asynchronous event sequences with time length 50 generated via a 5-dimensional Hawkes process. The exogenous base intensity of each event type is uniformly sampled from $[0, \frac{1}{5}]$. The sine-like impact functions are generated as

$$\phi_{cc'}(t) = \begin{cases} b_{cc'}(1 - \cos(\omega_{cc'}t - \pi s_{cc'})), & t \in [0, \frac{2-s_{cc'}}{4\pi\omega_{cc'}}], \\ 0, & \text{otherwise,} \end{cases} \quad c, c' = 1, \dots, 5,$$

where $\{b_{cc'}, \omega_{cc'}, s_{cc'}\}$ are set as $\{0.05, 0.6\pi, 1\}$ when $c, c' \in \{1, 2, 3\}$, $\{0.05, 0.4\pi, 0\}$ when $c, c' \in \{4, 5\}$, $\{0.02, 0.2\pi, 0\}$ when c (or c') = 4, c' (or c) $\in \{1, 2, 3\}$. The piecewise constant impact functions are the truncated results of above sine-like ones.

We test various learning algorithms on each of the two data sets with 10 trials. In each trial, $N = \{50, \dots, 250\}$ sequences are chosen randomly as training set while the remaining

250 sequences are chosen as testing set. In all trials, Gaussian basis functions are used, whose number and bandlimit are decided by Algorithm 2.

We test our algorithm with various parameters in a wide range, where $\alpha_P, \alpha_S, \alpha_G \in [10^{-2}, 10^4]$. The curves of *Loglike* w.r.t. the three parameters are shown in Fig. 2.3. We can find that the learning result is relatively stable when changing the parameters in a wide range. According to the experimental result, we set $\alpha_S = 10$, $\alpha_G = 100$, $\alpha_P = 1000$ as optimal configurations.

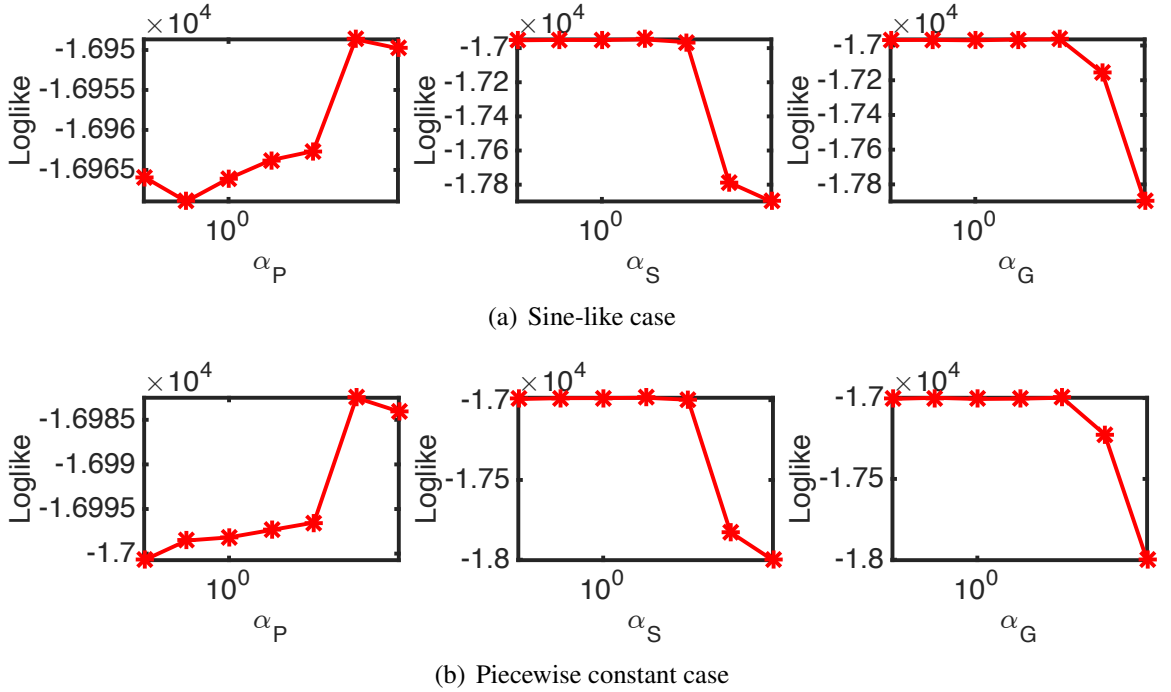


Figure 2.3: The curves of *Loglike* w.r.t. the change of α_P , α_G and α_S are shown. In each subfigure, left: $\alpha_G = 100$, $\alpha_S = 10$, $\alpha_P \in [10^{-2}, 10^4]$; middle: $\alpha_G = 100$, $\alpha_P = 1000$, $\alpha_S \in [10^{-2}, 10^4]$; right: $\alpha_P = 1000$, $\alpha_S = 10$, $\alpha_G \in [10^{-2}, 10^4]$.

In Fig. 2.4, we can find that our learning algorithm performs better than its competitors on both data sets w.r.t. various N (i.e., higher *Loglike*, lower e_μ and e_ϕ). Especially when having few training sequences, the ODE-based and the LS algorithm need to learn too many parameters from insufficient samples so they are inferior to our MLE-SGLP algorithm and its variants because of the over-fitting problem. By increasing the number of training sequences, the performance of the ODE-based algorithm does not improve a lot — the

nature of its non-convexity may lead the ODE-based algorithm to fall into local optimal. All MLE-based algorithms are superior to the ODE-based algorithm and the LS algorithm, and the proposed regularizers indeed help to improve the learning results. Specifically, if the clustering structure is available, our MLE-SGLP algorithm will obtain the best results. Otherwise, our MLE-SGL algorithm will be the best, which is slightly better than MLE-GL and MLE-S.

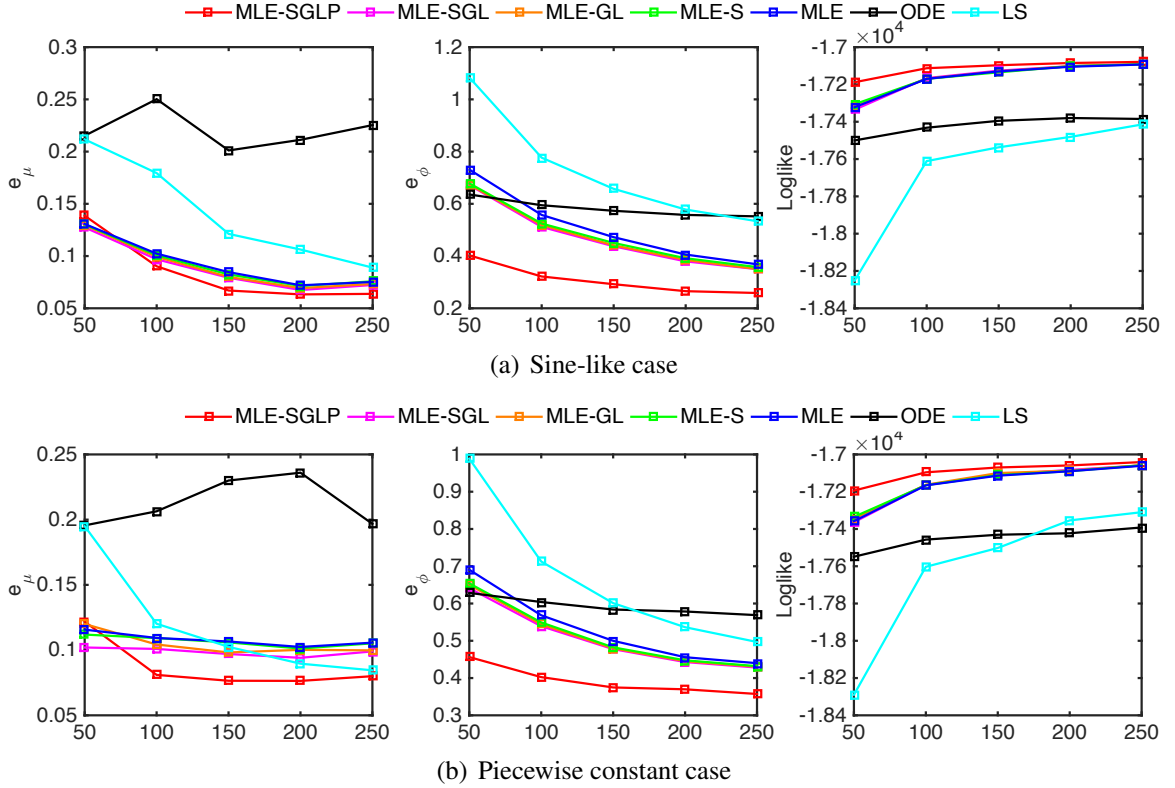
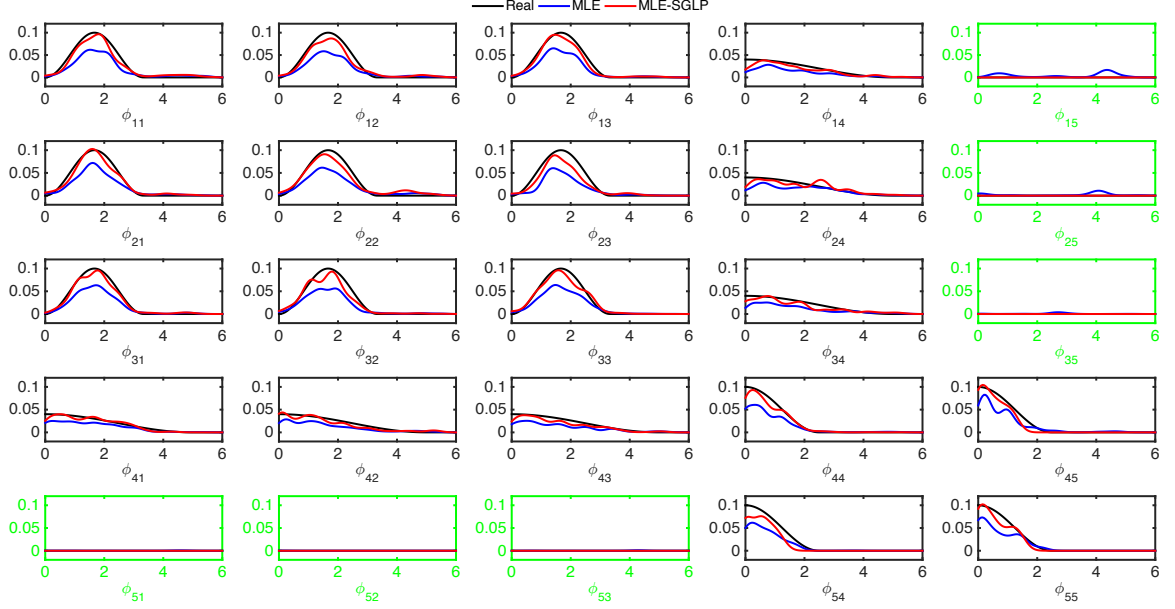
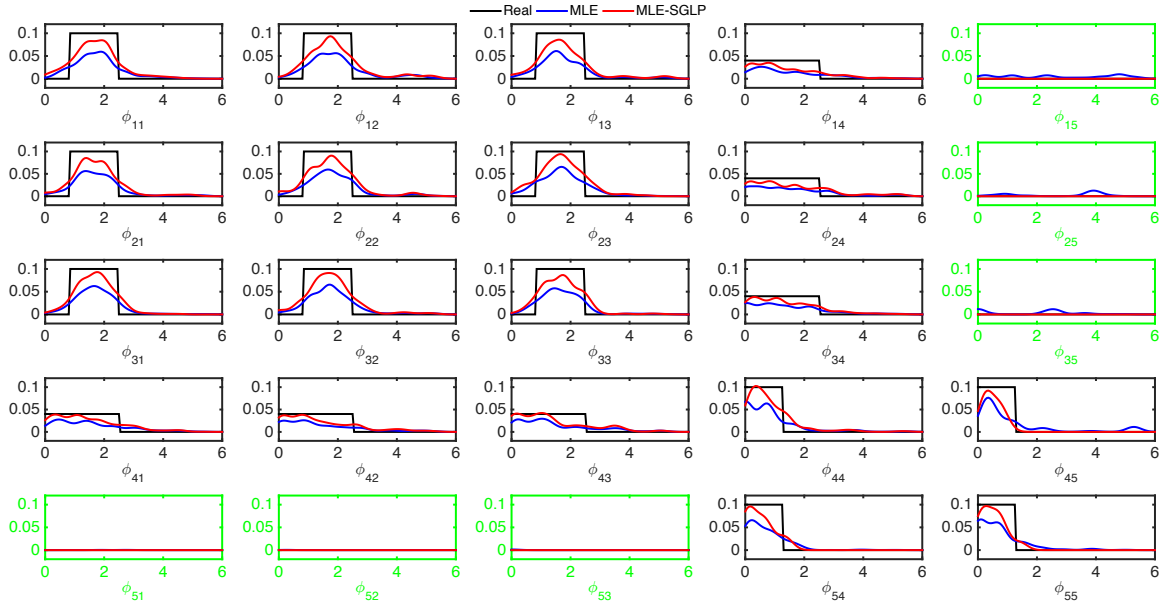


Figure 2.4: The e_μ , e_ϕ , and $Loglike$ for various methods.

To demonstrate the importance of the sparse-group-lasso regularizer, Fig. 2.5 visualizes the estimates of the impact functions obtained by various methods. The Granger causality graph of the target Hawkes process is learned by finding those all-zero impact functions (the green subfigures). Our MLE-SGLP algorithm obtains right all-zero impact functions while the pure MLE algorithm sometimes fails because of the lack of the sparse-group-lasso regularizer. This means that introducing sparse-group-lasso into the framework of the MLE is necessary for learning Granger causality. Note that, even if the basis functions we select



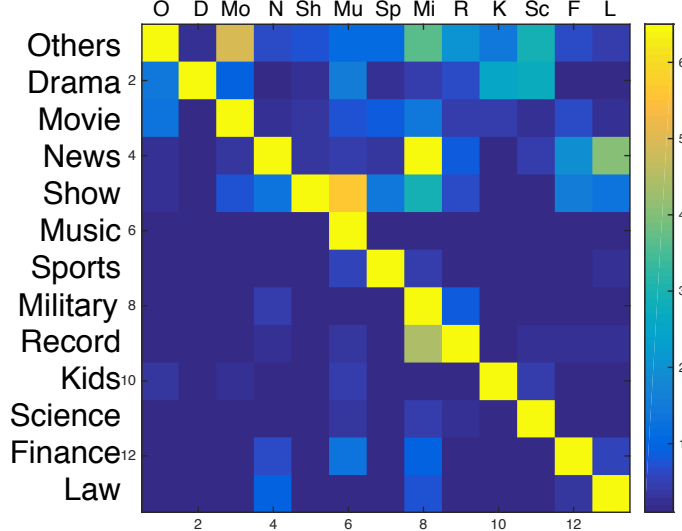
(a) Sine-like case



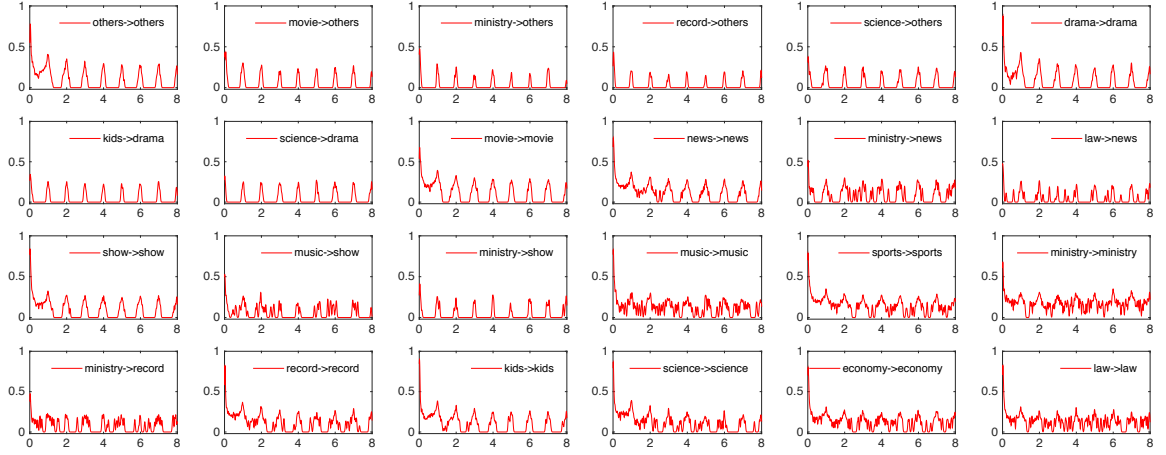
(b) Piecewise constant case

Figure 2.5: Contributions of regularizers: the comparisons of the impact functions obtained via MLE-SGLP and pure MLE. The green subfigures represent the all-zero impact functions. The black curves are real impact functions. The blue curves are the estimates based on pure MLE, and the red curves are the estimates based on our MLE-SGLP.

do not match well with the real case (i.e., the Gaussian basis functions are not suitable for piecewise constant impact functions), our algorithm can still learn the Granger causality graph of the Hawkes process robustly. As Fig. 2.4(b) shows, although the estimates of the



(a) Infectivity matrix



(b) Top 24 impact functions

Figure 2.6: (a) The infectivity matrix of TV programs. The element in the c -th row and the c' -th column is $\int_0^\infty \phi_{cc'}(s)ds$. (b) The estimates of the nonzero impact functions learned from the IPTV data. By ranking the infectivity $\int_0^\infty \phi_{cc'}(s)ds$ from high to low, the top 24 impact functions are shown. For the convenience of visualization, $\phi_{cc'}^{0.25}(t)$ is shown in each subfigure.

non-zero impact functions based on Gaussian basis functions do not fit the ground truth well, those all-zero impact functions are learned exactly via our MLE-SGLP algorithm.

2.5.2 Real-world Data

We test our algorithm on the IPTV viewing record data set [87, 22, 53]. The data set contains the viewing behavior of 7100 users (i.e., what and when they watch) in the IPTV system of Shanghai Telecom Inc. from January to November 2012. The TV programs are categorized into 13 classes, which have self-and mutually-triggering patterns. For example, the user watching an episode of a drama is likely to watch the following episodes (self-triggering) and the news of the actors appearing in the drama (mutually-triggering) in the future. Therefore, we model the users' viewing behaviors via a Hawkes process and learn the Granger causality among the categories.

In this case, the pairwise similarity regularizer is not applied because the clustering structure is not available. The training data is the viewing behaviors in the first 10 months and testing data is the viewing behaviors in the last month. Considering the fact that many TV programs are daily or weekly periodic and the time length of most TV programs is about 20-40 minutes, we set the time length of impact function to be 8 days (i.e., the influence of a program will not exist over a week) and the number of samples $M = 576$ (i.e., one sample per 20 minutes). The cut-off frequency of the sampling function is $\omega_0 = \pi M/T$, where T is the number of minutes in 8 days. Table 2.2 gives the *Loglike* obtained via various methods w.r.t. different training sequences. We can find that with the increase of the training data, all the methods have improvements. Compared with the ODE-based algorithm and pure MLE algorithm, the MLE with regularizers has better *Loglike* and our MLE-SGL algorithm obtains the best result, especially when the training set is small (i.e., the event sequences in one month). Note that here the LS algorithm doesn't work. Even using a PC with 16GB memory, the LS algorithm runs out-of-memory in this case because it requires to discretize long event sequences with dense samples.

We define the infectivity of the c' -th TV program category on the c -th one as $\int_0^\infty \phi_{cc'}(s)ds$, which is shown in Fig. 2.6(a). The collection of the infectivity corresponds to the adjacency matrix of the Granger causality graph. Additionally, by ranking the infectivity from high to

Table 2.2: The *Loglike* ($\times 10^6$) obtained via various methods

| ALG. | ODE | MLE | MLE-S | MLE-GL | MLE-SGL |
|-----------|--------|--------|--------|---------------|---------------|
| 1 MONTH | -2.066 | -1.904 | -1.888 | -1.885 | -1.880 |
| 4 MONTHS | -1.992 | -1.895 | -1.880 | -1.879 | -1.876 |
| 7 MONTHS | -1.957 | -1.882 | -1.877 | -1.874 | -1.873 |
| 10 MONTHS | -1.919 | -1.876 | -1.874 | -1.872 | -1.872 |

low, the top 24 impact functions are selected and shown in Fig. 2.6(b). Our algorithm works well because the following reasonable phenomena are observed in our learning results:

1) All the TV program categories have obvious self-triggering patterns because the TV programs display periodically. Users are likely to watch them daily at the same time. Our learning results reflect these phenomena: the main diagonal elements of the infectivity matrix in Fig. 2.6(a) are much larger than other ones, and the estimates of the impact functions in Fig. 2.6(b) have daily-periodic patterns.

2) The popular categories having a large number of viewers and long displaying time, e.g., “drama”, “movie”, “news” and “talk show”, are likely to be triggered by others, while the unpopular ones having relative fewer but fixed viewers and short displaying time, e.g., “music”, “kids’ program”, “science”, are mainly triggered by themselves. The infectivity matrix we learned reflects these patterns — the non-diagonal elements involving those unpopular categories are very small or zero. In Fig. 2.6(b) the non-zero impact functions mainly involve popular categories. Additionally, because few viewing events about these categories are observed in the training data, the estimates of the impact functions involving unpopular categories are relatively noisy.

In summary, our algorithm performs better on the IPTV data set than its competitors. The learning results are reasonable and interpretable, which prove the rationality and the feasibility of our algorithm.

2.6 Summary

In this chapter, we learn Granger causality for asynchronous event sequences with the help of Hawkes processes. Combining the MLE with the sparse-group-lasso, we propose an effective algorithm to learn the Granger causality graph with high accuracy. We demonstrate the robustness and the rationality of our work on both synthetic and real-world data.

CHAPTER 3

FEATURE-BASED CLUSTERING: MULTI-TASK MULTI-DIMENSIONAL HAWKES PROCESSES

3.1 Introduction

In the previous chapter, we learn Granger causality for asynchronous event sequences based on Hawkes process model. That work implicitly assumes that all event sequences are generated via a single Hawkes process. In practical information systems, however, event sequences are likely to have clustering structure, i.e., they are generated via different Hawkes processes and the event sequences exhibiting similar triggering patterns can be characterized as a cluster. Take the IPTV data in the previous chapter as an example. In an IPTV system, users buy and watch various TV programs, and their watching behaviors (when and what they watch) form a large number of asynchronous event sequences. The triggering patterns of TV program categories in the sequences reflect the watching preferences of the users. Furthermore, the users having similar preferences can be clustered according to the similarity of their triggering patterns.

Simultaneously learning the triggering pattern of event types within individual event sequence and the clustering structure across event sequences has great practical significance, because it enables both local and global depictions of the entire dynamic system. In this chapter, we propose a feature-based clustering method based on a novel point process model, namely multi-task multi-dimensional Hawkes process (MMHP). In particular, we apply MMHP to learn features of the event sequences (i.e., infectivity matrices of the event sequences), and then cluster the event sequences accordingly. As the event sequences in the same cluster should share implicit relatedness, we learn our MMHP model under the multi-task learning framework [88, 89]. We impose sparse and low-rank constraints on

the tensor of the infectivity matrices so as to induce the clustering structure of the event sequences, and propose an effective algorithm that takes advantage of alternating direction method of multipliers (ADMM), majorization minimization and Euler-Lagrange equations to solve the learning problem.

Our MMHP model considers both the triggering pattern of event types in each event sequence and the clustering structure of the event sequences. The triggering patterns in the event sequences and their similarities are captured by a structured infectivity tensor. With the help of the multi-task learning method, the learning algorithm of the MMHP can avoid over-fitting where only few events are observed in some event sequences — the structure of the infectivity tensor facilitates the knowledge transfer from event sequences with many observed events to those short of observations. We evaluate our method and compare it with other state-of-the-art methods. Our method obtains encouraging clustering results on both synthetic and real-world data.

3.2 Proposed Model

3.2.1 Multi-Task Multi-Dimensional Hawkes Processes

Given N asynchronous event sequences with C event types, denoted as $\mathcal{S} = \{\mathbf{s}_n = \{(t_i^n, c_i^n)\}_{i=1}^{I_n}\}_{n=1}^N$, we model each event sequence \mathbf{s}_n via a multi-dimensional Hawkes process. In particular, the conditional intensity function for \mathbf{s}_n on type- c event at time t is

$$\begin{aligned}\lambda_c^n(t) &= \mu_c^n + \sum_{j:t_j^n < t} \phi_{cc_j}^n(t - t_j^n) \\ &= \mu_c^n + \sum_{j:t_j^n < t} a_{cc_j}^n g(t - t_j^n).\end{aligned}\tag{3.1}$$

The model parameters in (3.1) fall in two categories:

- A *global* triggering kernel $g(t)$ shared via all event sequences reflects the attenuating influential effects from historical events.

- A *local* endogenous infectivity matrix $\mathbf{A}^n = [a_{cc'}^n]$ and a *local* exogenous intensity vector $\boldsymbol{\mu}^n = [\mu_1^n, \dots, \mu_C^n]^\top$ are specific to the event sequence \mathbf{s}_n , $n = 1, \dots, N$.

We represent such \mathbf{A}^n 's and $\boldsymbol{\mu}^n$'s into an endogenous infectivity tensor $\mathcal{A} = [a_{cc'}^n] \in \mathbb{R}_+^{C \times C \times N}$ and an exogenous intensity matrix $\boldsymbol{\mu} = [\mu_c^n] \in \mathbb{R}_+^{C \times N}$, respectively. Compared with the model in (2.1), here we follow the time-invariant assumption of impact function, but just use a single base $g(t)$ to represent each impact function (i.e., $\phi_{cc'}^n(t) = a_{cc'}^n g(t)$). Such a simplification enhances the scalability of model. As a result, each infectivity matrix \mathbf{A}^n actually is the adjacent matrix of the Granger causality graph of \mathbf{s}_n , which can be used as the feature to cluster event sequences.

3.2.2 Structural Constraints on MMHP

We impose structural constraints on the parameters in (3.1) so as to reveal the clustering structure of the event sequences. Specifically, we impose sparse and low-rank constraints on the flattened matrix $\mathbf{A} = [\text{vec}(\mathbf{A}^1), \dots, \text{vec}(\mathbf{A}^N)] \in \mathbb{R}^{C^2 \times N}$ from tensor \mathcal{A} . The sparsity constraint is based on the observations that typically within each sequence, only a subset of event types happens and triggers others. The intuition behind the low-rank constraint is to uncover the clustering structure of the event sequences according to the similarity among their infectivity matrices \mathbf{A}^u 's. Here we assume that the similarity is described by the self-representation property of the infectivity matrices, which implies the low-rank structure of \mathbf{A} . These two structural constraints are reasonable for practical systems. For example, in an IPTV system, each user typically has a preference over a small number of program categories, and such a preference can be common among a large number of users.

3.2.3 Learning Task

Based on our MMHP model and the structural constraints of the model's parameter, we can learn our MMHP model via combining MLE with sparse and low-rank regularizers. Specifically, we learn the parameters $\Theta = \{\boldsymbol{\mu}, \mathbf{A}, g(t)\}$ by solving the following optimiza-

tion problem:

$$\min_{\Theta \geq 0} -\log \mathcal{L}(\mathcal{S}; \Theta) + \alpha \mathcal{R}(g) + \lambda_1 \|\mathbf{A}\|_* + \lambda_2 \|\mathbf{A}\|_1, \quad (3.2)$$

where $\log \mathcal{L}(\mathcal{S}; \Theta)$ is the log-likelihood function that can be written as:

$$\begin{aligned} \log \mathcal{L}(\mathcal{S}; \Theta) &= \sum_{n=1}^N \left(\sum_{i=1}^{I_n} \log \lambda_{c_i^n}^n(t_i^n) - \sum_{c=1}^C \int_{T_b^n}^{T_e^n} \lambda_c^n(t) dt \right) \\ &= \sum_{n=1}^N \left[\sum_{i=1}^{I_n} \log \left(\mu_{c_i^n}^n + \sum_{j: t_j^n < t_i^n} a_{c_i^n c_j^n}^n g(t_i^n - t_j^n) \right) \right. \\ &\quad \left. - (T_e^n - T_b^n) \sum_{c=1}^C \mu_c^n - \sum_{c=1}^C \sum_{i=1}^{I_n} a_{cc_i^n}^n \int_{T_b^n}^{T_e^n - t_i^n} g(t) dt \right], \end{aligned} \quad (3.3)$$

The second term $\mathcal{R}(g)$ in (3.2), defined as

$$\mathcal{R}(g) = \int_0^\infty [g'(t)]^2 dt, \quad (3.4)$$

regularizes the smoothness of the triggering kernel [30] (i.e., $\int_0^\infty [g'(t)]^2 dt < \infty$ and $g'(t)$ is the first-order derivative of $g(t)$). The nuclear norm and the ℓ_1 norm in (3.2) impose the low-rank and the sparsity on \mathbf{A} .

3.3 Learning Algorithm

3.3.1 Learning MMHP

We apply the ADMM method [90, 91, 34] to learn our MMHP model. In particular, we introduce two auxiliary variables $\{\mathbf{Z}_1, \mathbf{Z}_2\}$ and two dual variables $\{\mathbf{U}_1, \mathbf{U}_2\}$ to (3.2), and then rewrite the optimization problem as

$$\begin{aligned} \min_{\Theta \geq 0} & -\log \mathcal{L}(\mathcal{S}; \Theta) + \alpha \mathcal{R}(g) + \lambda_1 \|\mathbf{Z}_1\|_* + \lambda_2 \|\mathbf{Z}_2\|_1 + \rho(\text{tr}(\mathbf{U}_1^\top (\mathbf{A} - \mathbf{Z}_1))) \\ & + \rho(\text{tr}(\mathbf{U}_2^\top (\mathbf{A} - \mathbf{Z}_2))) + \frac{\rho}{2} (\|\mathbf{A} - \mathbf{Z}_1\|_F^2 + \|\mathbf{A} - \mathbf{Z}_2\|_F^2). \end{aligned} \quad (3.5)$$

We solve the problem in (3.5) via alternating optimization and update \mathbf{A} and $\boldsymbol{\mu}$, $g(t)$, $\{\mathbf{Z}_1, \mathbf{Z}_2\}$, and $\{\mathbf{U}_1, \mathbf{U}_2\}$ iteratively.

Step 1: Update \mathbf{A} and $\boldsymbol{\mu}$. We first update \mathbf{A} and $\boldsymbol{\mu}$ by a majorization-minimization algorithm. Denote the parameters of the k -th iteration as $\boldsymbol{\Theta}^{(k)}$. We find a surrogate function of objective function with the help of Jensen's inequality:

$$\begin{aligned} \mathcal{Q}(\boldsymbol{\Theta}; \boldsymbol{\Theta}^{(k)}) = & \sum_{n=1}^N \left[\sum_{i=1}^{I_n} \left(p_{ii}^n \log \frac{\mu_{c_i^n}^n}{p_{ii}^n} + \sum_{j=1}^{i-1} p_{ij}^n \log \frac{a_{c_i^n c_j^n}^n g(\tau_{ij}^n)}{p_{ij}^n} \right) - (T_e^n - T_b^n) \sum_{c=1}^C \mu_c^n \right. \\ & \left. - \sum_{c=1}^C \sum_{i=1}^{I_n} \int_{T_e^n}^{T_b^n - t_i^n} \left((a_{cc_i^n}^n)^2 \frac{g^{(k)}(t)}{2a_{cc_i^n}^{n(k)}} + (g(t))^2 \frac{a_{cc_i^n}^{n(k)}}{2g^{(k)}(t)} \right) dt \right] \\ & + \frac{\rho}{2} (\|\mathbf{A} - \mathbf{Z}_1^{(k)} + \mathbf{U}_1^{(k)}\|_F^2 + \|\mathbf{A} - \mathbf{Z}_2^{(k)} + \mathbf{U}_2^{(k)}\|_F^2) + \alpha \mathcal{R}(g). \end{aligned} \quad (3.6)$$

where $\tau_{ij}^n = t_i^n - t_j^n$, and

$$p_{ii}^n = \frac{\mu_{c_i^n}^{n(k)}}{\mu_{c_i^n}^{n(k)} + \sum_{j=1}^{i-1} a_{c_i^n c_j^n}^{n(k)} g(t_i^n - t_j^n)}, \quad p_{ij}^n = \frac{a_{c_i^n c_j^n}^{n(k)} g(t_i^n - t_j^n)}{\mu_{c_i^n}^{n(k)} + \sum_{j=1}^{i-1} a_{c_i^n c_j^n}^{n(k)} g(\tau_{ij}^n)}.$$

Considering the terms related to $\{\boldsymbol{\mu}, \mathbf{A}\}$ and solving the equations $\frac{\partial \mathcal{Q}}{\partial \mu_c^n} = 0$ and $\frac{\partial \mathcal{Q}}{\partial a_{cc'}^n} = 0$, we obtain closed-form solutions to $\boldsymbol{\mu} = [\mu_c^n]$ and $\mathbf{A} = [a_{cc'}^n]$:

$$\mu_c^{n(k+1)} = \frac{\sum_{i:c_i^n=c} p_{ii}^n}{T_e^n - T_b^n}, \quad a_{cc'}^{n(k+1)} = \frac{-B + \sqrt{B^2 - 4AC}}{2A}, \quad (3.7)$$

$$\begin{aligned} A = & 2\rho a_{cc'}^{n(k)} + \sum_{i:c_i^n=c'} \int_{T_b^n}^{T_e^n - t_i^n} g(t) dt, \\ B = & \rho a_{cc'}^{n(k)} (u_{1,cc'}^{n(k)} - z_{1,cc'}^{n(k)} + u_{2,cc'}^{n(k)} - z_{2,cc'}^{n(k)}), \\ C = & -a_{cc'}^{n(k)} \sum_{i:c_i^n=c} \sum_{j:j < i, c_j^n=c'} p_{ij}^n, \end{aligned}$$

where $z_{cc'}^{n(k)}$'s and $u_{cc'}^{n(k)}$'s are the elements in $\mathbf{Z}^{n(k)}$ and $\mathbf{U}^{n(k)}$ w.r.t. $a_{cc'}^n$'s, respectively.

Step 2: Update $g(t)$. The triggering kernel g can be updated in an infinite dimen-

sional space [30]. The optimization of $\mathcal{Q}(\Theta; \Theta^{(k)})$ w.r.t. $g(t)$ is equivalent to minimize $\int_0^\infty f(g, g')dt$, where

$$\begin{aligned} f(g, g') = & - \sum_{n=1}^N \sum_{i=1}^{I_n} \sum_{j=1}^{i-1} p_{ij}^n \log g(t) \mathbb{I}(t = t_i^n - t_j^n) \\ & + \sum_{n=1}^N \sum_{c=1}^C \sum_{i=1}^{I_n} \frac{g^2(t) a_{cc_i}^{n(k)}}{2g^{(k)}(t)} \mathbb{I}(t \leq T_e^n - t_i^n) + \alpha[g'(t)]^2, \end{aligned} \quad (3.8)$$

where $\mathbb{I}(\cdot)$ is the indicator function which returns 1 if the input predicate is true and 0 otherwise. The solution of $g(t)$ satisfies Euler-Lagrange equation:

$$\frac{\partial f}{\partial g} - \frac{d}{dt} \frac{\partial f}{\partial g'} = 0 \rightarrow -\frac{D(t)}{g(t)} + F(t)g(t) - 2\alpha g''(t) = 0, \quad (3.9)$$

where $g''(t)$ is the second-order derivative of $g(t)$ and

$$\begin{aligned} F(t) &= \sum_{n=1}^N \sum_{c=1}^C \sum_{i=1}^{I_n} \frac{a_{cc_i}^{n(k)}}{g^{(k)}(t)} \mathbb{I}(t < T_e^n - t_i^n), \\ D(t) &= \sum_{n=1}^N \sum_{i=1}^{I_n} \sum_{j=1}^{i-1} p_{ij}^n \mathbb{I}(t = t_i^n - t_j^n). \end{aligned}$$

We solve (3.9) numerically using the following efficient Seidel-type iterations. Setting the sampling interval as Δt , we discretize the differential equation over small intervals $m\Delta t$ for $m = 1, \dots, M$:

$$-\frac{D_m}{g_m} + F_m g_m - 2\alpha \frac{g_{m+1} + g_{m-1} - 2g_m}{\Delta t^2} = 0, \quad (3.10)$$

where $g_m = g(m\Delta t)$, $F_m = F(m\Delta t)$, and $D_m = D(m\Delta t)$. M is the number of samples of g and $M\Delta t$ is the length of g . For $m = 1, \dots, M$, we can update g_m via fixing the remaining $\{g_{m'}\}_{m' \neq m}$ and solving the above quadratic equation.

Step 3: Update Z_1 and Z_2 . Given the $A^{(k+1)}$ obtained via step 1, we update Z_1 by

solving the following optimization problem:

$$\min_{\mathbf{Z}_1} \lambda_1 \|\mathbf{Z}_1\|_* + \rho(\text{tr}((\mathbf{U}_1^{(k)})^\top (\mathbf{A}^{(k+1)} - \mathbf{Z}_1))) + \frac{\rho}{2} \|\mathbf{A}^{(k+1)} - \mathbf{Z}_1\|_F^2.$$

The solution is obtained by shrinking the singular values of $\mathbf{A}^{(k+1)} + \mathbf{U}_1^{(k)}$:

$$\mathbf{Z}_1^{(k+1)} = E_{\lambda_1/\rho}(\mathbf{A}^{(k+1)} + \mathbf{U}_1^{(k)}), \quad (3.11)$$

where $E_{\lambda_1/\rho}(\cdot)$ is the soft-thresholding function of matrix's singular values with the threshold λ_1/ρ [34]. Similarly, we update \mathbf{Z}_2 by solving the following optimization problem:

$$\min_{\mathbf{Z}_2} \lambda_2 \|\mathbf{Z}_2\|_1 + \rho(\text{tr}((\mathbf{U}_2^{(k)})^\top (\mathbf{A}^{(k+1)} - \mathbf{Z}_2))) + \frac{\rho}{2} \|\mathbf{A}^{(k+1)} - \mathbf{Z}_2\|_F^2,$$

and the solution is

$$\mathbf{Z}_2^{(k+1)} = S_{\lambda_2/\rho}(\mathbf{A}^{(k+1)} + \mathbf{U}_2^{(k)}), \quad (3.12)$$

where $S_{\lambda_2/\rho}(\cdot)$ is the soft-thresholding function of matrix's elements. The threshold is λ_2/ρ .

Step 4: Update \mathbf{U}_1 and \mathbf{U}_2 . The dual variables \mathbf{U}_1 and \mathbf{U}_2 are updated as follows:

$$\mathbf{U}_1^{(k+1)} = \mathbf{U}_1^{(k)} + (\mathbf{A}^{(k+1)} - \mathbf{Z}_1^{(k+1)}), \quad \mathbf{U}_2^{(k+1)} = \mathbf{U}_2^{(k)} + (\mathbf{A}^{(k+1)} - \mathbf{Z}_2^{(k+1)}). \quad (3.13)$$

The whole learning algorithm is summarized in Algorithm 3.

3.3.2 Clustering Based on Infectivity Tensor

After learning the MMHP model from event sequences, we regard the \mathbf{A} as the features of the event sequences and cluster them accordingly. Various traditional clustering methods like K-means [92, 93] and Gaussian mixture model can be utilized.

Algorithm 3 MMHP Learning Algorithm

```
1: Input: event sequences  $\mathcal{S}$ , parameters  $\{\lambda_1, \lambda_2, M, \alpha\}$ 
2: Output:  $\mathbf{A}$ ,  $\boldsymbol{\mu}$  and  $g(t)$ 
3: Initialize  $\boldsymbol{\mu} \in \mathbb{R}_+^{C \times N}$ ,  $\mathbf{A} \in \mathbb{R}_+^{C^2 \times N}$ ,  $g \in \mathbb{R}_+^M$  randomly.
4:  $\mathbf{Z}_1^{(0)} = \mathbf{Z}_2^{(0)} = \mathbf{A}^{(0)}$ ,  $\mathbf{U}_1^{(0)} = \mathbf{U}_2^{(0)} = \mathbf{0}$ ,  $k = 0$ .
5: repeat
6:    $k = k + 1$ 
7:   Update  $\{\boldsymbol{\mu}^{(k)}, \mathbf{A}^{(k)}\}$  by (3.7)
8:   repeat
9:     for  $m = 1 : M$ 
10:      Update  $g_m^{(k)}$  by (3.10)
11:    end
12:   until convergence
13:   Update  $\mathbf{Z}_1^{(k)}, \mathbf{Z}_2^{(k)}$  by (3.11) and (3.12)
14:   Update  $\mathbf{U}_1^{(k)}, \mathbf{U}_2^{(k)}$  by (3.13)
15: until convergence
16:  $\mathbf{A} = \mathbf{A}^{(k)}$ ,  $\boldsymbol{\mu} = \boldsymbol{\mu}^{(k)}$ ,  $g(t) = \{g_m^{(k)} | m = 1 : M\}$ 
```

3.4 Experimental Results

We evaluate the performance of MMHP on both synthetic and real-world data. Specifically, we compare our learning algorithm (MMHP) with the following alternatives:

- The Full: the infectivity tensor \mathcal{A} has no structural constraints (i.e., $\lambda_1 = \lambda_2 = 0$ in (3.2)). This method is similar to that in [30].
- The Sparse: only the sparsity constraint is imposed on \mathcal{A} (i.e., $\lambda_1 = 0$ in (3.2)).
- The LowRank: only the low-rank constraint is imposed on \mathcal{A} (i.e., $\lambda_2 = 0$ in (3.2)).

We use the following metrics to evaluate the performance of various methods:

- *LogLike*: the log-likelihood of testing data using the trained model.
- e_ϕ : the averaged estimation error of the impact functions $\{a_{cc'}^n g(t)\}$

$$e_\phi = \frac{1}{NC^2} \sum_{n=1}^N \sum_{c,c'=1}^C \int_0^\infty [a_{cc'}^n g(t) - \hat{a}_{cc'}^n \hat{g}(t)]^2 dt,$$

where $\{\hat{\mathbf{A}}, \hat{g}(t)\}$ represents the real parameters and $\{\mathbf{A}, g(t)\}$ represents the corresponding estimates.

- *RankCorr*: the averaged Kendall's rank correlation coefficient between the columns of the real $\hat{\mathbf{A}}$ and those of the estimated \mathbf{A} .
- *ClusAcc*: the clustering accuracy, defined as the percentage of sequences clustered correctly based on the learned infectivity tensor. It is only available for synthetic data.
- *ClusDiff*: a metric of clustering accuracy for real-world data, where the ground truth of parameters and the clustering indices are unavailable. In specific, we first cluster sequences by applying K-means [92, 93] on the estimated \mathbf{A} . After computing centers of clusters from estimated \mathbf{A} , we then construct a clustered infectivity tensor $\tilde{\mathbf{A}}$, where the n -th slide $\tilde{\mathbf{A}}^n$ is the center of the cluster the \mathbf{A}^n belongs to. Denote the set of possible methods as \mathcal{I} (Here $\mathcal{I} = \{\text{Full, Sparse, LowRank, MMHP}\}$). For a certain method, *ClusDiff* measures the difference of the log-likelihood calculated from the ever best performing method and the log-likelihood calculated using the constructed $\tilde{\mathbf{A}}$ from that method, that is, ClusDiff of the i -th method is defined as

$$ClusDiff(i) = \max_{j \in \mathcal{I}} \log \mathcal{L}(\mathcal{S}; \mathbf{A}_j) - \log \mathcal{L}(\mathcal{S}; \tilde{\mathbf{A}}_i). \quad (3.14)$$

If a clustering result is good, then each cluster center should be representative for capturing the dynamics of the sequences belonging to the cluster, and the difference of log-likelihood caused by replacing the specific infectivity matrices with the cluster centers should be small.

3.4.1 Synthetic Data

We generate a synthetic data set in which there are 5 event types. The sequences are generated so as to fall into two clusters with equal size. We generate the flatten version of the infectivity tensor as $\mathbf{A} = [\text{thres}(\mathbf{u}_1 \mathbf{v}_1^\top), \text{thres}(\mathbf{u}_2 \mathbf{v}_2^\top)]$, where $\mathbf{u}_1, \mathbf{u}_2 \in \mathbb{R}_+^{C^2}$ and $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}_+^{N/2}$ are four random vectors with values uniformly distributed over $[0, 1]$. The function $\text{thres}(\mathbf{X})$ randomly sets half of the rows in \mathbf{X} as zero. In this way, the generated \mathbf{A} is low-rank and sparse, and inherently represents two clusters. We generate an intensity matrix $\boldsymbol{\mu}$ from a uniform distribution over $[0, 0.001]$. We use an exponential kernel $g(t) = \exp(-t)$ with $t \in (0, 20]$ as the triggering kernel. Given the above parameters, we simulate 100 training sequences and 100 testing sequences respectively. Each event sequence contains 500 events.

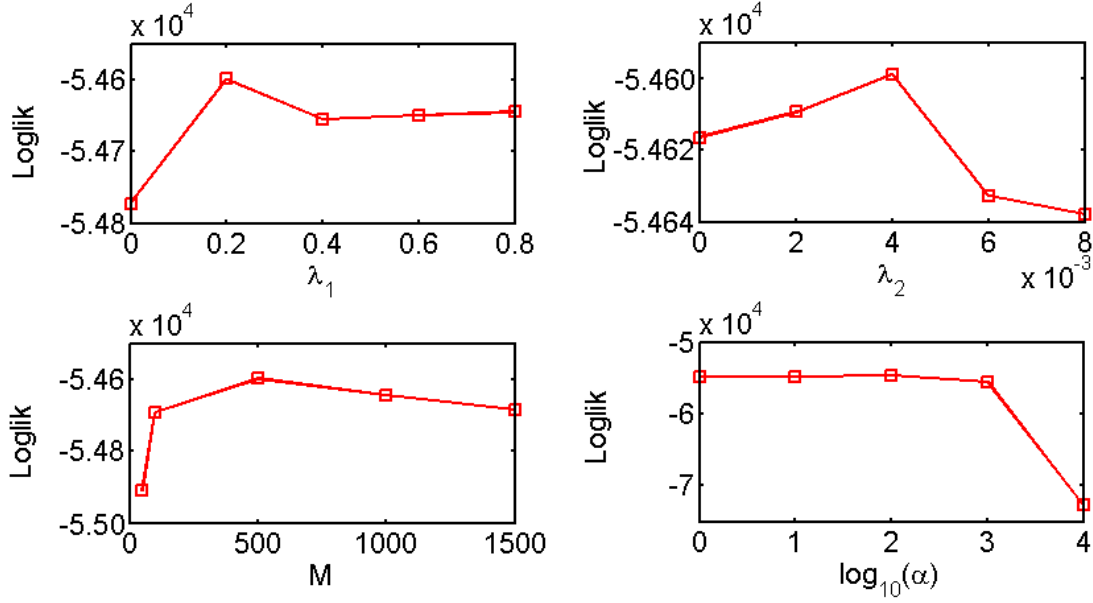


Figure 3.1: Parameter study for the MMHP.

We conduct a parameter study on the four parameters $\{\lambda_1, \lambda_2, M, \alpha\}$ involved in our MMHP algorithm. The parameters λ_1 and λ_2 control the rank and the sparsity level of the infectivity tensor, and M and α control the sampling rate and the smoothness of the triggering kernel. We first identify the optimal parameter configuration by doing a

grid search in the parameter space of $\lambda_1 \in [0, 0.8]$, $\lambda_2 \in [0, 0.008]$, $M \in [50, 1500]$ and $\alpha \in [1, 10000]$. With the identified optimal configuration as $\lambda_1 = 0.2$, $\lambda_2 = 0.004$, $M = 500$, $\alpha = 100$, we fix 3 parameters as their optimal values each time and alter the fourth parameter to train a different MMHP model. Fig. 3.1 represents the *LogLike* of such MMHP w.r.t. the different parameters, which demonstrates the robustness of our algorithm w.r.t. its parameters. In particular, as the value of λ_1 or λ_2 grows larger, the *LogLike* of the MMHP first increases and then decreases. This demonstrates the effectiveness of λ_1 and λ_2 in controlling the rank and the sparsity of the infectivity tensor, respectively. A similar trend applies for M — extremely small M leads to a coarse estimation of the triggering kernel while extremely large M leads to over-fitting. The performance is relatively stable for $\alpha \in [1, 1000]$. The performance degrades a lot only in the case using too large α , which leads to an over-smoothed triggering kernel.

We compare the MMHP with the Full, the Sparse and the LowRank methods on *LogLike*, e_ϕ and *RankCorr*, respectively. The parameters of these algorithms are configured according to the result of the parameter study above¹. Fig. 3.2 presents the averaged results of the 10 trials. It shows that the MMHP consistently achieves significantly better performance (i.e., higher *LogLike*, lower e_ϕ and higher *RankCorr*) than other methods on different training sets. We cluster the training event sequences via applying K-means clustering algorithm on the infectivity tensors obtained via various methods. and evaluate the clusters using *ClusAcc*. The results in Table 3.1 demonstrate that the MMHP outperforms others in uncovering structures across sequences and the performance difference is more significant when training sequences have fewer events. The experimental results further verify the effectiveness of MMHP in preventing over-fitting.

¹The λ_1 , λ_2 in the Full, the λ_1 in the Sparse, and the λ_2 in the LowRank are fixed to be 0's.

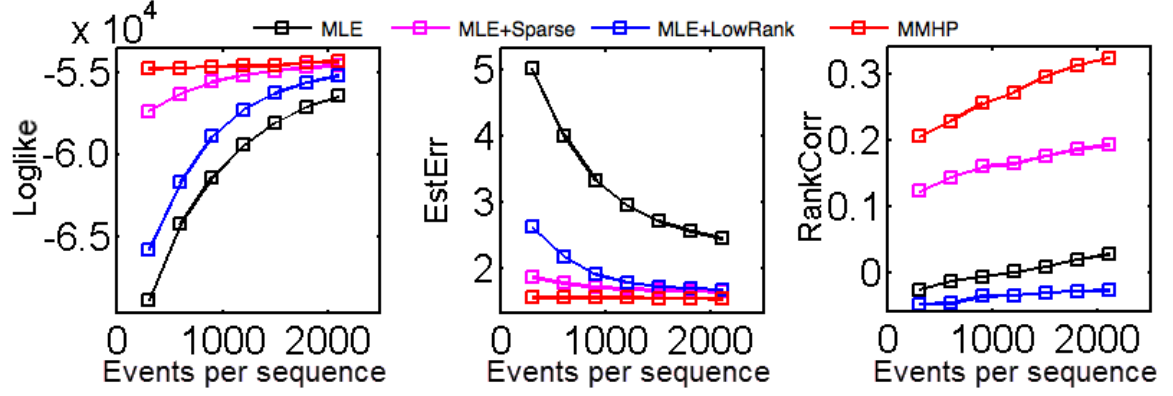


Figure 3.2: Experimental results on synthetic data.

Table 3.1: Clustering on Synthetic Data ($ClusAcc (\times 100\%)$).

| #Events per sequences | Full | Sparse | LowRank | MMHP |
|-----------------------|------|--------|---------|-------------|
| 50 | 0.67 | 0.81 | 0.80 | 0.88 |
| 75 | 0.86 | 0.92 | 0.90 | 0.94 |
| 100 | 0.97 | 0.97 | 0.98 | 0.99 |

Bold numbers correspond to the best performance.

3.4.2 Real-world Data

we apply the MMHP to model the IPTV viewing data [87]. The training set contains 2967 users' viewing records in 11 months and 9000 TV programs are categorized into 25 classes. Given the event sequences of the users, we learn models using the MMHP, the Full, the Sparse and the LowRank, respectively. Similar to the work in the previous chapter, we set the length of the triggering kernel as 11520 minutes (8 days) and the sampling interval Δt as 20 minutes ($M = 576$) for all the learning algorithms. Such a configuration ensures that 1) for daily and weekly TV programs, the triggering kernel will capture their periodic influence on its own, and 2) for most TV programs, which are 20-40 minutes in length, the triggering kernel has a good resolution to capture the influence from a previous watching event.

We learn the models from the first L months and test them on the data of the $L + 1$ -th month. Running the experiments in the same way as for the synthetic data, we obtain the averaged $LogLike$ w.r.t. $L = 2, \dots, 10$ shown in Fig. 3.3(a). Similar to the results on the

synthetic data, the MMHP produces better *LogLike* than other methods. Additionally, we visualize the triggering kernel learned by the MMHP in Fig. 3.3(b), which clearly shows that the triggering kernel captures the temporal influence decay of a program on its following programs, which reveals the periodic nature of user watching behaviors well. In particular, there are 8 spikes in the kernel with 1 spike per day periodically, which corresponds to the self-triggering pattern of daily programs. The first spike is the highest and corresponds to the influence of previous watching events on that day. It indicates that the mutual-triggering patterns among various program categories mainly exist in the watching behaviors happening in the same day. With time elapsing, the intensity of the spikes is reduced gradually, which corresponds to the decay of influences over time. However, the spike of the 7-th day is a little higher than those of its adjacent days, indicating the existence of the self-triggering pattern of weekly programs.

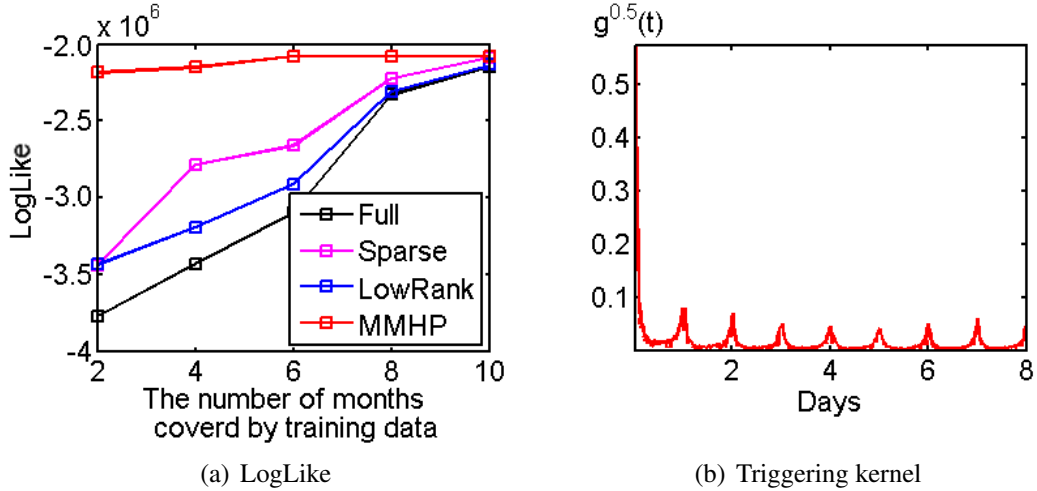


Figure 3.3: Experimental results on an IPTV system.

Furthermore, we evaluate the performance of various methods on clustering users using *ClusDiff*. For each method, we cluster the users into 5 clusters via applying K-means to the learned infectivity tensor. Table 3.2 shows that compared with the other methods, the MMHP not only achieves the best learning result ($\max \log \mathcal{L}(\mathcal{S}; \mathbf{A}))$ given the data of the first 11 months, but also obtains the smallest *ClusDiff*. This means that the clustering

results learned via our method are reasonable —the clustering centers are representative, which are able to reflect users’ viewing preferences.

Table 3.2: Clustering Performance on IPTV Data ($\times 10^7$).

| Metric | Full | Sparse | LowRank | MMHP |
|---------------------------------------|-------|--------|---------|--------------|
| $\max \text{LogLike}(\mathcal{A})$ | — | — | — | -1.02 |
| $\text{LogLike}(\tilde{\mathcal{A}})$ | -1.78 | -1.18 | -1.41 | -1.09 |
| ClusDiff | 0.76 | 0.16 | 0.39 | 0.07 |

Bold numbers correspond to the best performance.

3.5 Summary

In this chapter, we propose a multi-task multi-dimensional Hawkes process model and the corresponding learning algorithm for clustering asynchronous event sequences. The proposed clustering method actually extends the Granger causality analysis work in the previous chapter — learning adjacent matrices of the event sequences’ Granger causality graphs jointly as a infectivity tensor and clustering the event sequences based on the tensor. Taking advantage of multi-task learning methods, we impose the sparse and low-rank structure of the infectivity tensor, the proposed model captures the triggering pattern of event types within each event sequence and the clustering structure across different event sequences jointly. The proposed method has superior performance compared to its competitors on both synthetic data and real-world data.

CHAPTER 4

MODEL-BASED CLUSTERING: DIRICHLET MIXTURE MODEL OF HAWKES PROCESS

4.1 Introduction

In the previous chapter, we propose an event sequence clustering method based on multi-task multi-dimensional Hawkes processes. Similar to other feature-based clustering methods [7], our method first constructs features for various event sequences and then clusters the sequences based on the features. The feature-based clustering methods assign diverse parameters to event sequences individually, rather than clusters, which provide us with models having high capacity and high degree of freedom. However, the clustering results of the feature-based methods are highly dependent on the quality of the learned or pre-defined features, which may not be robust to the case having few training samples (i.e., extremely short event sequences). Even if we add some structural constraints to regularize the models, as we do in the previous chapter, the learning results are still likely to be over-fitting. What is worse, the number of parameters increases linearly w.r.t. the number of event sequences. In large-scale applications, these feature-based clustering method would suffer to the problem of scalability. Additionally, how to decide the number of clusters for asynchronous event sequences? Whether and when are the clusters identifiable? These important problems of asynchronous event sequence clustering are difficult to solve in the framework of the feature-based methods because they are highly dependent on the results of feature extraction and selection.

Based on the analysis above, it is urgent for us to go beyond the feature-based clustering methods and solve the practical and theoretical problems mentioned above. To make concrete progress, we model asynchronous event sequences based on multiple Hawkes pro-

cesses and propose a model-based clustering method. In particular, we propose a Dirichlet Mixture model of Hawkes Processes (DMHP for short), where each event sequence belonging to a cluster is modeled via a specific Hawkes process, different clusters have different Hawkes processes accordingly, and the prior distribution of the Hawkes processes is decided via a Dirichlet process. Similar to Chapter 2, we apply a basis representation of Hawkes processes to enhance the flexibility of the model.

We propose a variational Bayesian inference algorithm to learn the DMHP model effectively, avoiding the time-consuming sampling step in the Markov chain Monte Carlo (MCMC) approach [33]. The convergence of our learning algorithm is investigated, and an adaptive inner-outer iteration strategy is applied to the method based on open-loop control theory. The computational complexity of our learning method is analyzed in details. Furthermore, we prove the local identifiability of our DMHP model in both theoretical and empirical ways. Following the work in [94, 95], we visualize the “edges of identifiability” for various clustering methods (including ours) with different numbers of samples. Experiments on both synthetic and real-world data show that our method indeed achieves superior clustering results compared with its competitors on clustering purity and consistency.

The novelties and the advantages of our model include:

- 1) We propose a model-based clustering method for asynchronous event sequences, learning clustering structures directly from the event sequences themselves. Our model represents the event sequences within a cluster via just one Hawkes process such that these event sequences can leverage information with each other in the learning phase. The proposed model requires few parameters and is robust to the problems like over-fitting and model misspecification.
- 2) We propose an effective learning algorithm based on variational inference. Dirichlet processes help us to learn the number of clusters automatically. An adaptive inner-outer iteration strategy is explored based on open-loop control theory, which accelerates the convergence of our learning algorithm greatly and achieves a trade-

off between convergence performance and computational complexity.

- 3) The local identifiability of our model is proven, and the influence of sample complexity on the identifiability is studied. To our knowledge, it is the first systematical research on the identifiability problem of asynchronous event sequence clustering task.

4.2 Proposed Model and Learning Algorithm

In this section, we introduce our Dirichlet mixture model of Hawkes processes in details, capturing the temporal dynamic within each sequence and the clustering structure across various sequences simultaneously. We propose a variational inference algorithm for learning the model effectively.

4.2.1 Dirichlet Mixture Model of Hawkes Processes

Given a set of event sequences $\mathcal{S} = \{\mathbf{s}_n\}_{n=1}^N$, where $\mathbf{s}_n = \{(t_i^n, c_i^n)\}_{i=1}^{I_n}$ contains a series of events $c_i^n \in \mathcal{C} = \{1, \dots, C\}$ and their time stamps $t_i^n \in [0, T_n]$, we assume that the event sequences belong to K clusters, which are generated via K time-invariant Hawkes processes. According to the definition of Hawkes process's intensity function in (2.1), for the event sequence belonging to the k -th cluster, its conditional intensity function corresponding to the type- c event is represented as

$$\lambda_c^k(t) = \mu_c^k + \sum_{t_i < t} \phi_{cc_i}^k(t - t_i) = \mu_c^k + \sum_{t_i < t} \sum_{d=1}^D a_{cc_id}^k g_d(t - t_i). \quad (4.1)$$

Here $\boldsymbol{\mu}^k = [\mu_c^k] \in \mathbb{R}_+^C$ is the exogenous base intensity of the k -th Hawkes process. Each impact function $\phi_{cc'}^k(t)$ has a basis representation $\sum_d a_{cc'd}^k g_d(t - t_i)$, where $g_d(t) \geq 0$ is the d -th basis function and $\mathbf{A}^k = [a_{cc'd}^k] \in \mathbb{R}_{0+}^{C \times C \times D}$ is the coefficient tensor. Here \mathbb{R}_+ is the domain of positive real number and \mathbb{R}_{0+} is the domain of nonnegative real number.

Using a probabilistic mixture model, we represent the probability of an arbitrary event sequence \mathbf{s} as

$$p(\mathbf{s}; \Theta) = \sum_{k=1}^K \pi^k p(\mathbf{s}|k),$$

$$p(\mathbf{s}|k) = \text{HP}(\mathbf{s}|\boldsymbol{\mu}^k, \mathbf{A}^k) = \prod_i \lambda_{c_i}^k(t_i) \exp \left(- \sum_c \int_0^T \lambda_c^k(s) ds \right). \quad (4.2)$$

Here π^k is the distribution of the clusters and $\text{HP}(\mathbf{s}|\boldsymbol{\mu}^k, \mathbf{A}^k)$ is the conditional probability of the event sequence \mathbf{s} given the k -th Hawkes process, which actually is the likelihood function of \mathbf{s} .

According to the Bayesian graphical model, we regard the parameters of the Hawkes processes as random variables obeying certain distributions. For $\boldsymbol{\mu}^k$'s, we consider its positiveness and assume that they are samples of $C \times K$ independent Rayleigh distributions. For \mathbf{A}^k 's, we consider its nonnegativeness and sparsity (indicated via [34, 22, 29]) and assume that they are samples of $C \times C \times D \times K$ independent exponential distributions. Furthermore, we use a Dirichlet process with hyper-parameter $\boldsymbol{\alpha}$ to generate the prior distribution of the clusters. Mathematically, we can describe the proposed Dirichlet mixture model of Hawkes processes in a generative way as

$$\begin{aligned} \boldsymbol{\pi} &\sim \text{Dir}(\boldsymbol{\alpha}), \\ k|\boldsymbol{\pi} &\sim \text{Category}(\boldsymbol{\pi}), \\ \boldsymbol{\mu} &\sim \text{Rayleigh}(\mathbf{B}), \\ \mathbf{A} &\sim \text{Exp}(\boldsymbol{\Sigma}), \\ \mathbf{s}|k, \boldsymbol{\mu}, \mathbf{A} &\sim \text{HP}(\boldsymbol{\mu}_k, \mathbf{A}_k). \end{aligned}$$

Here $\boldsymbol{\mu} = [\mu_c^k] \in \mathbb{R}_+^{C \times K}$ obeys independent Rayleigh distributions with a hyper-parameter matrix $\mathbf{B} = [\beta_c^k]$. $\mathbf{A} = [a_{cc'd}^k] \in \mathbb{R}_{0+}^{C \times C \times D \times K}$ obeys independent exponential distributions with a hyper-parameter tensor $\boldsymbol{\Sigma} = [\sigma_{cc'd}^k]$. $\text{HP}(\boldsymbol{\mu}_k, \mathbf{A}_k)$ represents the Hawkes process

generating s .

Denote the latent variables indicating the labels of clusters as a matrix $\mathbf{Z} \in \{0, 1\}^{N \times K}$. We can factorize the joint distribution of all variables $p(\mathbf{S}, \mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\mu}, \mathbf{A})$ as¹

$$\begin{aligned}
& p(\mathbf{S}|\mathbf{Z}, \boldsymbol{\mu}, \mathbf{A})p(\mathbf{Z}|\boldsymbol{\pi})p(\boldsymbol{\pi})p(\boldsymbol{\mu})p(\mathbf{A}), \\
& p(\mathbf{S}|\mathbf{Z}, \boldsymbol{\mu}, \mathbf{A}) = \prod_{n,k} \text{HP}(\mathbf{s}_n|\boldsymbol{\mu}^k, \mathbf{A}^k)^{z_{nk}}, \\
& p(\mathbf{Z}|\boldsymbol{\pi}) = \prod_{n,k} (\pi^k)^{z_{nk}}, \\
& p(\boldsymbol{\pi}) = \text{Dir}(\boldsymbol{\pi}|\boldsymbol{\alpha}), \\
& p(\boldsymbol{\mu}) = \prod_{c,k} \text{Rayleigh}(\mu_c^k|\beta_c^k), \\
& p(\mathbf{A}) = \prod_{c,c',d,k} \text{Exp}(a_{cc'd}^k|\sigma_{cc'd}^k).
\end{aligned} \tag{4.3}$$

Note that our mixture model of Hawkes processes are different from the models in [96, 19, 56]. Those models focus on the sub-sequence clustering problem within an event sequence. Each event in a sequence is generated via a mixture model of Hawkes processes, whose intensity is a weighted sum of multiple intensity functions from different Hawkes processes. Our model, however, aims at finding the clusters of event sequences. The intensity of each event is generated via a single Hawkes process, while the likelihood of an event sequence is a mixture of the likelihood functions of different Hawkes processes.

4.2.2 Variational Bayesian Learning Algorithm

Learning (4.3) directly via the maximum likelihood estimation is intractable. Instead of purely using MCMC-based learning method like [33], we propose an effective variational inference algorithm implemented as nested expectation-maximization. Specifically, we consider a variational distribution which factorizes between the latent variables and the

¹Rayleigh($x|\beta$) = $\frac{x}{\beta^2} e^{-\frac{x^2}{2\beta^2}}$, Exp($x|\sigma$) = $\frac{1}{\sigma} e^{-\frac{x}{\sigma}}$, $x \geq 0$.

parameters so that

$$q(\mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\mu}, \mathbf{A}) = q(\mathbf{Z})q(\boldsymbol{\pi}, \boldsymbol{\mu}, \mathbf{A}). \quad (4.4)$$

An nested EM algorithm with inner-outer iterations can be used to optimize (B.1).

Update Responsibility (E-step). The logarithm of the optimized factor $q^*(\mathbf{Z})$ is approximated as

$$\begin{aligned} & \log q^*(\mathbf{Z}) \\ &= \mathbb{E}_{\boldsymbol{\pi}}[\log p(\mathbf{Z}|\boldsymbol{\pi})] + \mathbb{E}_{\boldsymbol{\mu}, \mathbf{A}}[\log p(\mathbf{S}|\mathbf{Z}, \boldsymbol{\mu}, \mathbf{A})] + \mathbb{C} \\ &= \sum_{n,k} z_{nk} (\mathbb{E}[\log \pi^k] + \mathbb{E}[\log \text{HP}(\mathbf{s}_n|\boldsymbol{\mu}^k, \mathbf{A}^k)]) + \mathbb{C} \\ &= \sum_{n,k} z_{nk} \left(\mathbb{E}[\log \pi^k] + \mathbb{E}[\sum_i \log \lambda_{c_i}^k(t_i) - \sum_c \int_0^{T_n} \lambda_c^k(s) ds] \right) + \mathbb{C} \\ &\approx \underbrace{\sum_{n,k} z_{nk} \left(\mathbb{E}[\log \pi^k] + \sum_i \left(\log \mathbb{E}[\lambda_{c_i}^k(t_i)] - \frac{\text{Var}[\lambda_{c_i}^k(t_i)]}{2\mathbb{E}^2[\lambda_{c_i}^k(t_i)]} \right) - \sum_c \mathbb{E}[\int_0^{T_n} \lambda_c^k(s) ds] \right)}_{\rho_{nk}} + \mathbb{C}. \end{aligned}$$

where \mathbb{C} is a constant and $\text{Var}[\cdot]$ represents the variance of random variable. Following the work in [97], we approximate each term $\mathbb{E}[\log \lambda_c^k(t)]$ via its second-order Taylor expansion $\log \mathbb{E}[\lambda_c^k(t)] - \frac{\text{Var}[\lambda_c^k(t)]}{2\mathbb{E}^2[\lambda_c^k(t)]}$. Then, the responsibility r_{nk} is calculated as

$$r_{nk} = \mathbb{E}[z_{nk}] = \frac{\rho_{nk}}{\sum_j \rho_{nj}}. \quad (4.5)$$

Denote $N_k = \sum_n r_{nk}$ for all k 's.

Update Parameters (M-step). The logarithm of optimal factor $q^*(\boldsymbol{\pi}, \boldsymbol{\mu}, \mathbf{A})$ is

$$\begin{aligned} & \log q^*(\boldsymbol{\pi}, \boldsymbol{\mu}, \mathbf{A}) \\ &= \sum_k \log(p(\boldsymbol{\mu}^k)p(\mathbf{A}^k)) + \mathbb{E}_{\mathbf{Z}}[\log p(\mathbf{Z}|\boldsymbol{\pi})] + \log p(\boldsymbol{\pi}) + \sum_{n,k} r_{nk} \log \text{HP}(\mathbf{s}_n|\boldsymbol{\mu}^k, \mathbf{A}^k) + \mathbb{C}. \end{aligned}$$

We can estimate the parameters of Hawkes processes via:

$$\hat{\boldsymbol{\mu}}, \hat{\mathbf{A}} = \arg \max_{\boldsymbol{\mu}, \mathbf{A}} \log(p(\boldsymbol{\mu})p(\mathbf{A})) + \sum_{n,k} r_{nk} \log \text{HP}(\mathbf{s}_n | \boldsymbol{\mu}^k, \mathbf{A}^k). \quad (4.6)$$

Following the work in [34, 19, 29], we need to apply an EM algorithm to solve (B.5) iteratively, which corresponds to the *inner iterations* in the M-step. Specifically, we initialize $\boldsymbol{\mu}$ and \mathbf{A} via the expectations of their distributions used in the E-step (i.e., $\boldsymbol{\mu} = \sqrt{\frac{\pi}{2}}\mathbf{B}$ and $\mathbf{A} = \boldsymbol{\Sigma}$). Again, applying Jensen's inequality, we obtain the surrogate function of the objective function:

$$\begin{aligned} & \log(p(\boldsymbol{\mu})p(\mathbf{A})) + \sum_{n,k} r_{nk} \log \text{HP}(\mathbf{s}_n | \boldsymbol{\mu}^k, \mathbf{A}^k) \\ &= \sum_{c,k} \left[\log \mu_c^k - \frac{1}{2} \left(\frac{\mu_c^k}{\beta_c^k} \right)^2 \right] - \sum_{c,c',d,k} \frac{a_{cc'd}^k}{\sigma_{cc'd}^k} \\ & \quad + \sum_{n,k} r_{nk} \left[\sum_i \log \lambda_{c_i}^k(t_i) - \sum_c \int_0^{T_n} \lambda_c^k(s) ds \right] \\ &\geq \sum_{c,k} \left[\log \mu_c^k - \frac{1}{2} \left(\frac{\mu_c^k}{\beta_c^k} \right)^2 \right] - \sum_{c,c',d,k} \frac{a_{cc'd}^k}{\sigma_{cc'd}^k} \\ & \quad + \sum_{n,k} r_{nk} \left[\sum_i \left(p_{ii}^k \log \frac{\mu_{c_i}^k}{p_{ii}^k} + \sum_{j < i, d} p_{ijd}^k \log \frac{a_{c_i c_j d}^k g_d(\tau_{ij})}{p_{ijd}^k} \right) \right. \\ & \quad \left. - \sum_c T_n \mu_c^k - \sum_{c,i,d} a_{ccid}^k G_d(T_n - t_i) \right] \\ &= Q, \end{aligned}$$

where $p_{ii}^k = \frac{\mu_{c_i}^k}{\lambda_{c_i}^k(t_i)}$, and $p_{ijd}^k = \frac{a_{c_i c_j d}^k g_d(\tau_{ij})}{\lambda_{c_i}^k(t_i)}$. Setting $\frac{\partial Q}{\partial \mu_c^k} = 0$ and $\frac{\partial Q}{\partial a_{cc'd}^k} = 0$, we have

$$\hat{\mu}_c^k = \frac{-B + \sqrt{B^2 - 4AC}}{2A}, \quad \hat{a}_{cc'd}^k = \frac{\sum_n r_{nk} \sum_{i:c_i=c} \sum_{j:c_j=c'} p_{ijd}^k}{1/\sigma_{cc'd}^k + \sum_n r_{nk} \sum_{i:c_i=c'} G_d(T_n - t_i)}. \quad (4.7)$$

where

$$A = \frac{1}{(\beta_c^k)^2}, \quad B = \sum_n r_{nk} T_n, \quad C = -1 - \sum_n r_{nk} \sum_{i:c_i=c} p_{ii}^k.$$

After repeating several inner iterations, we can get optimal $\hat{\boldsymbol{\mu}}$, $\hat{\mathbf{A}}$, and $N_k = \sum_n r_{nk}$, and update distributions as

$$\alpha^k = \alpha + N_k, \boldsymbol{\Sigma}^k = \hat{\mathbf{A}}^k, \mathbf{B}^k = \sqrt{2/\pi} \hat{\boldsymbol{\mu}}^k, \quad (4.8)$$

where $\boldsymbol{\alpha} = [\alpha, \dots, \alpha] \in \mathbb{R}^K$ is the initial parameter of the Dirichlet process. Repeating the E-step and the M-step above, our algorithm maximizes the log-likelihood function (i.e., the logarithm of (4.3)) and achieves optimal $\{\boldsymbol{\alpha}, \boldsymbol{\Sigma}, \mathbf{B}\}$ accordingly.

Update The Number of Clusters K . When the number of clusters K is unknown, we initialize K randomly and update it in the learning phase. There are multiple methods to update the number of clusters. Regarding our Dirichlet distribution as a finite approximation of a Dirichlet process, we set a large initial K as the truncation level. A simple empirical method is discarding the empty cluster (i.e., $N_k = 0$) and merging the cluster with N_k smaller than a threshold N_{min} in the learning phase. Besides this, we can apply the MCMC in [98, 99] to update K via merging or splitting clusters.

Repeating the three steps above, our algorithm maximizes the log-likelihood function (i.e., the logarithm of (4.3)) and achieves optimal $\{\boldsymbol{\Sigma}, \mathbf{B}\}$ accordingly. The details of our algorithm are given in Appendix B.

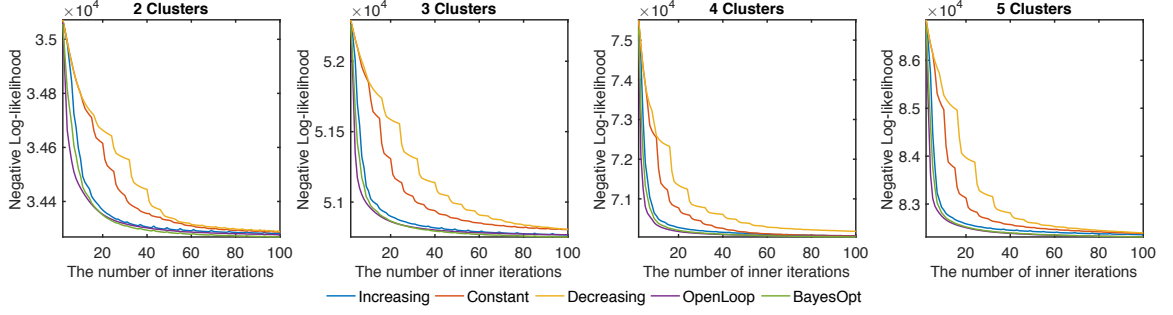
4.2.3 Adaptive Inner-Outer Iteration Strategies

The derivation above reveals that we decompose a complicated non-convex optimization problem into two convex subproblems (i.e., the E-step and the M-step) and implement our algorithm via a nested inner-outer iteration strategy, where the inner iteration corresponds to the M-step while the outer iteration corresponds to the loop of E-step and M-step. Obviously, the runtime of our algorithm is linearly proportional to the total number of inner iterations. More importantly, both the final achievable log-likelihood and convergence behavior of the algorithm highly depend on how we allocate the number of inner iterations

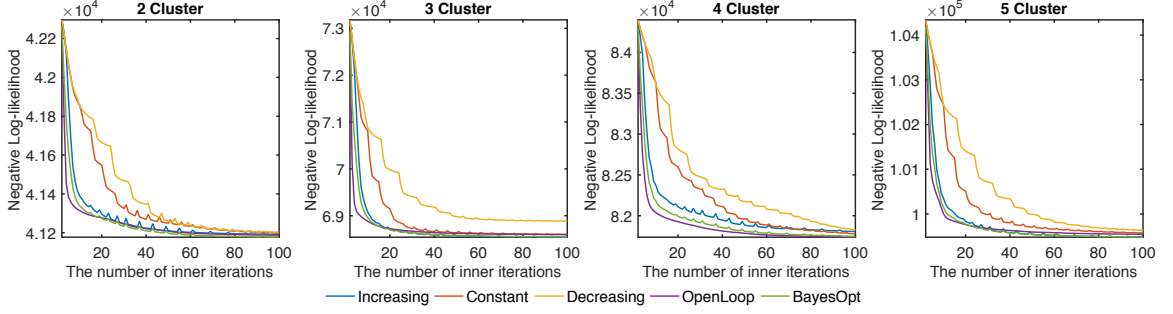
across the outer iterations. In this section, we investigate several allocation strategies and compare their performance on synthetic event sequences. Specifically, we test the following three strategies.

- First, the simplest method is *heuristic strategy*, which increases (or decreases) the number of inner iterations as the outer iteration progresses. Compared with constant inner iteration strategy (i.e., using constant number of inner iterations per outer iteration), such a strategy might improve the convergence of algorithm [100].
- The second method is *open-loop control* [101]: in each inner iteration we compute objective function via two methods respectively — updating parameters directly (i.e., just M-step) or first updating responsibilities and then updating parameters (i.e., a complete loop of E-step and M-step). The parameters corresponding to the smaller objective function are preserved for next iteration.
- The third method is *Bayesian optimization* [102, 103], which takes the number of inner iterations per outer iteration as the hyper-parameter of the model and optimizes it via maximizing the Expected Improvement (EI).

We apply these three strategies on two synthetic data sets and visualize their impacts on the convergence of the algorithm. Both of these two data sets generated via a mixture model of several (i.e., 2-5) 5-dimensional Hawkes processes. Each model has a sparse tensor \mathbf{A} . In the first data set, the nonzero elements in \mathbf{A} are distributed randomly, while in the second data set, each slide of \mathbf{A}^k , $k = 1, \dots, K$, contain several all-zero columns and rows (i.e. blockwise sparse tensor). The convergence curves obtained via various strategies are shown in Fig. 4.1, which demonstrate that the open-loop control and the Bayesian optimization obtain comparable performance on the convergence of algorithm, which outperform heuristic strategies consistently w.r.t. the structure of parameters and the number of clusters. It should be noted that although adjusting the number of inner iterations via different methodologies, both the open-loop control and the Bayesian optimization tend



(a) Random Sparse Infectivity Matrices



(b) Blockwise Sparse Infectivity Matrices

Figure 4.1: Comparison for various inner-outer iteration strategies on different synthetic data sets. The testing data set contains 2-5 clusters, and each cluster contains 200 event sequences generated via a 5-dimensional Hawkes process. Each convergence curve is the average of 5 trials' results. For the fairness, 20 outer iterations are applied in each trial. For heuristic strategy, the increasing method increases the number of inner iterations from 2 to 8; the decreasing method decreases the number from 8 to 2. The constant method fixes the number to 5.

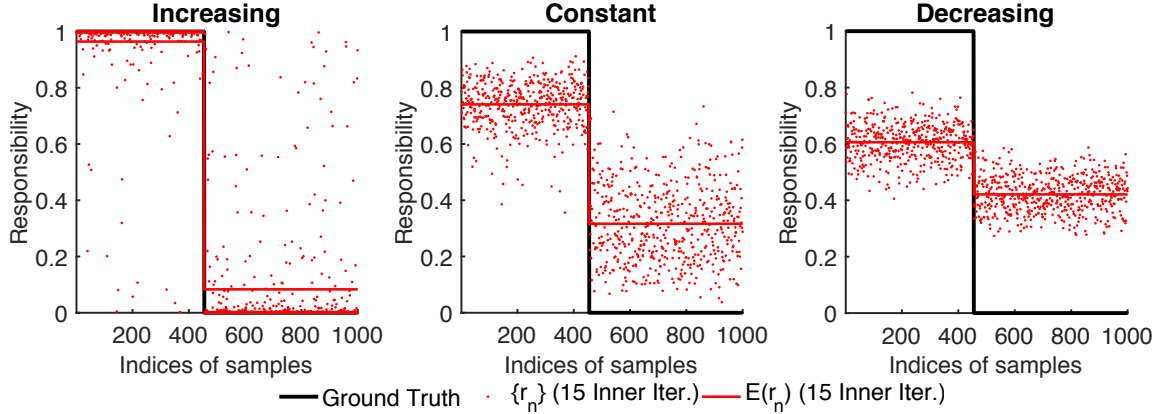


Figure 4.2: For each method, the ground truth of responsibility is shown as a black line, the responsibility after 15 inner iterations is shown as red dot, and its mean is shown as red line. Using increasing strategy, the responsibility reaches the ground truth more quickly.

to increase the number of inner iterations w.r.t. the number of outer iterations. In the beginning of our learning algorithm, the open-loop control strategy updates responsibilities frequently while the Bayesian optimization strategy similarly assigns small number of inner iterations. As a result, these two strategies reduce the negative log-likelihood more rapidly in the beginning and reach lower value finally than the heuristic strategies do.²

One explanation for this phenomenon is the non-convexity of our learning task. In the beginning of our learning algorithm, the estimated responsibility is not reliable. Too many inner iterations at that time might make learning results fall into bad local optimums. With the help of few inner iterations our algorithm can revise the estimation of responsibility frequently in the initial phase, so that learning results can jump out of bad local optimums easily and converge to good result quickly. Fig. 4.2 further verifies our explanation. Using “increasing” strategy, except for several outliers, the responsibilities of the samples belonging to a same cluster converge to the ground truth with high confidence after just 15 inner iterations, because the responsibilities has been revised over 5 times in this case. On the contrary, the responsibilities corresponding to the “constant” and the “decreasing” strategies have more uncertainty — the responsibilities are around 0.5 and far from the ground truth.

Based on the analysis above, the increasing allocation strategy indeed improves the convergence of our algorithm, and the open-loop control and the Bayesian optimization are superior to other competitors. Because the computational complexity of the open-loop control is much lower than that of the Bayesian optimization, in the following experiments, we apply open-loop control strategy to our learning algorithm. In summary, we give the scheme of our variational inference algorithm in Algorithm 4.

²The heuristic strategy that increasing the number of inner iterations follows the same tendency, and therefore, is just slightly worse than the open-loop control and the Bayesian optimization.

Algorithm 4 Learning DMHP

- 1: **Input:** $S = \{s_n\}_{n=1}^N$, the maximum number of clusters K , the maximum number of iteration I .
 - 2: **Output:** Optimal parameters of model, $\hat{\alpha}$, $\hat{\Sigma}$, and \hat{B} .
 - 3: Initialize α , Σ , B and $[r_{nk}]$ randomly, $i = 0$.
 - 4: **repeat**
 - 5: **Just M-step:**
 - 6: Given $[r_{nk}]$, update $\{\hat{\mu}^{(1)}, \hat{A}^{(1)}\}$ via (B.5), calculate negative log-likelihood $L^{(1)}$.
 - 7: **A loop of E-step and M-step:**
 - 8: Given $\{\alpha, \Sigma, B\}$, update responsibility via (B.3), denoted as $[r_{nk}^2]$.
 - 9: Given $[r_{nk}^2]$, update $\{\hat{\mu}^{(2)}, \hat{A}^{(2)}\}$ via (B.5), calculate negative log-likelihood $L^{(2)}$.
 - 10: **If** $L^{(1)} < L^{(2)}$
 - 11: Given $\{\hat{\mu}^{(1)}, \hat{A}^{(1)}\}$, update Σ, B via (B.6).
 - 12: **Else**
 - 13: Update $[r_{nk}]$ via $[r_{nk}^{(2)}]$.
 - 14: Given $[r_{nk}], \hat{\mu}^{(2)}, \hat{A}^{(2)}$, update α, Σ, B via (B.6).
 - 15: **End**
 - 16: Update the number of clusters via heuristic truncation or the MCMC method.
 - 17: $i = i + 1$.
 - 18: **until** $i = I$
 - 19: $\hat{\alpha} = \alpha$, $\hat{\Sigma} = \Sigma$, and $\hat{B} = B$.
-

4.3 Further Analysis

4.3.1 Local Identifiability

One of the most important questions of our mixture model is whether it is identifiable or not. According to the definition of Hawkes process, we can prove that our model is locally identifiable.

Theorem 4.3.1. *When the time of observation goes to infinite, the mixture model of the Hawkes processes defined in (4.2), in which the parameters of each Hawkes process is denoted as $\theta = \{\mu, A\} \in \mathbb{R}_+^C \times \mathbb{R}_{0+}^{C \times C \times D}$, is locally identifiable, i.e., for each parameter point $\Theta = \text{vec} \left(\begin{bmatrix} \pi^1 & \cdots & \pi^K \\ \theta^1 & \cdots & \theta^K \end{bmatrix} \right)$, there exists an open neighborhood of Θ containing no other Θ' which makes $p(s; \Theta) = p(s; \Theta')$ holds for all possible s .*

Proof. (sketched version) The information matrix $I(\Theta)$ is defined as

$$I(\Theta) = \mathbb{E}_s \left[\frac{\partial \log p(\mathbf{s}; \Theta)}{\partial \Theta} \frac{\partial \log p(\mathbf{s}; \Theta)}{\partial \Theta^\top} \right] = \mathbb{E}_s \left[\frac{1}{p^2(\mathbf{s}; \Theta)} \frac{\partial p(\mathbf{s}; \Theta)}{\partial \Theta} \frac{\partial p(\mathbf{s}; \Theta)}{\partial \Theta^\top} \right],$$

To each vector $\frac{\partial p(\mathbf{s}; \Theta)}{\partial \Theta}$ w.r.t. a parameter point Θ , there does not exist a nonzero vector of constants \mathbf{w} such that $\mathbf{w}^\top \frac{\partial p(\mathbf{s}; \Theta)}{\partial \Theta} = 0$ for all event sequence \mathbf{s} . According to the theorem in [104], in this case the information matrix is positive definite, and thus, is nonsingular. Then, according to the definition and the theorem in [105], we know that each Θ is a regular point of $I(\Theta)$ and locally identifiable. This completes the proof. \square

The detailed proof is given in Appendix C. Furthermore, we investigate the sample complexity of our method, which reflects the identifiability of our model in practice (i.e., when the number of samples is finite). We demonstrate the superiority of our method to its competitors via finding the edge of identifiability [95]. Specifically, we consider a binary clustering problem of Hawkes processes with 500 event sequences. For the k -th cluster, $k = 1, 2$, N_k event sequences are simulated via a 1-dimensional Hawkes processes with parameter θ^k . Taking the parameter of Hawkes process as the clustering center, we can calculate the distance between clustering centers, denoted as $d = \|\theta^1 - \theta^2\|_2$. Here, we make $N_1 + N_2 = 500$ and $N_1 < N_2$, and call the cluster having N_1 sequences “minor” cluster. The sample percentage of the minor cluster is denoted as $\pi^1 = \frac{N_1}{N_1 + N_2}$. Applying our DMHP model and its learning algorithm to the data generated under different d ’s and π^1 ’s, we can calculate the F1 scores of the minor cluster w.r.t. $\{d, \pi\}$. The high F1 score means that the minor cluster is identified with high accuracy while the low F1 score means that the minor cluster is hard to be identified. Fig. 4.3(a) visualizes the map of F1 scores w.r.t. the number of events per sequence. We can find that the F1 score obtained via our clustering method is close to 1 in most situations, which demonstrates the identifiability of our model in practice. The unidentifiable cases happen only in the following two situations: the parameters of different clusters are nearly equal (i.e., $d \rightarrow 0$); or the minor cluster is

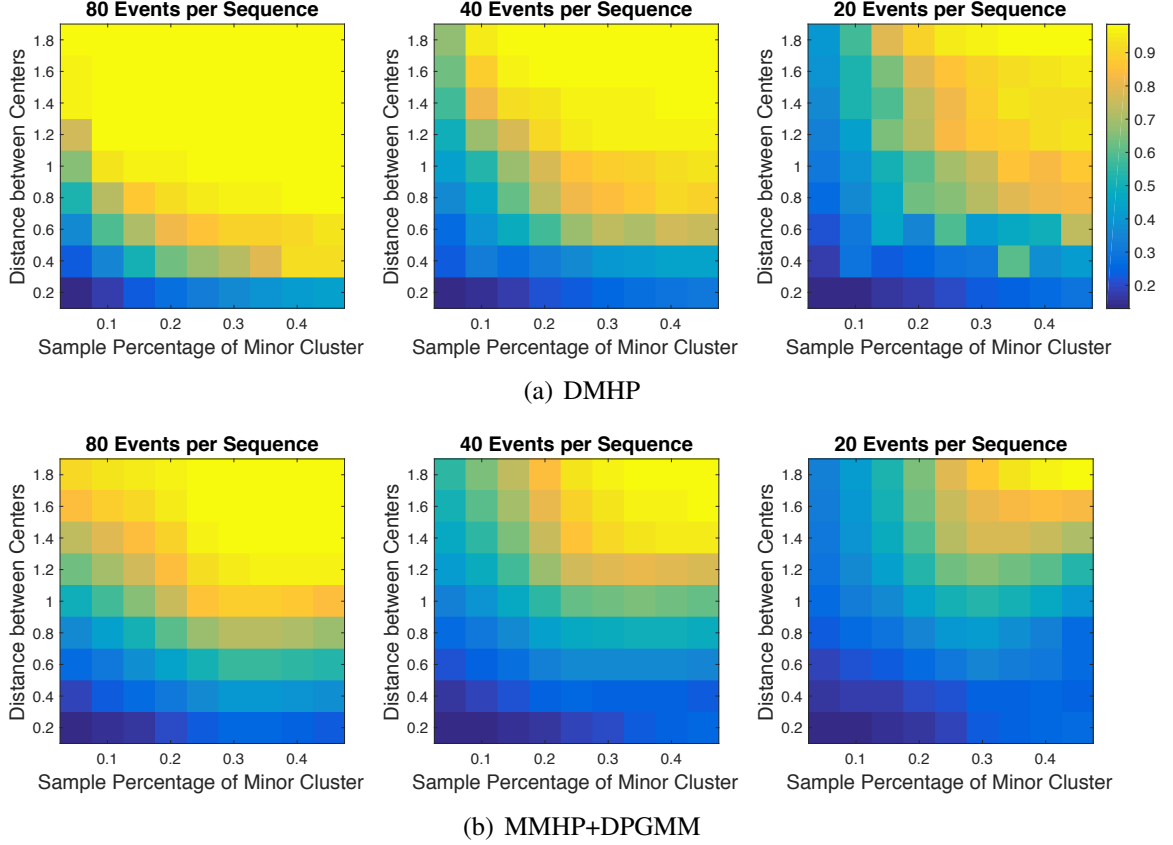


Figure 4.3: Comparison for various methods on F1 score of minor cluster with various number of events per sequence.

extremely small (i.e., $\pi^1 \rightarrow 0$). The boundary between the identifiable part (the yellow part) and the unidentifiable part (the blue part) is called the edge of identifiability. With the reducing of events per sequence, the unidentifiable part becomes larger and larger, which indicates that longer observation time is required.

For demonstrating the superiority of our model, we compare our method with its competitor on the area of identifiable part. The main competitor of our clustering method is the feature-based sequential data clustering methods in [10]. These methods assume that each event sequence yields to a parametric model, and then, they learn the parameters of the models as the features for clustering. A typical example is the multi-task multi-dimensional Hawkes process (MMHP) model introduced in the previous chapter, which learns a Hawkes process for each event sequence and clusters the event sequences according to their parameters. If ignoring the information of time stamps and treating event

sequences as time series, simpler models can be applied, e.g., the Markov chain (MC) and the vector auto-regressive (VAR) model. In that case, we can use the transition matrices of the event sequences as features. To cluster event sequences generated via Hawkes processes, we can first learn the MMHP model and then apply the Dirichlet processes Gaussian mixture model (DPGMM) [51, 106]. The F1 scores of the minor cluster are shown in Fig. 4.3(b). We can easily find that given the same data, the unidentifiable part obtained via the MMHP+DPGMM method is much larger than that of our method.

4.3.2 Computational Complexity and Acceleration

Suppose that we have N training sequences generated via C -dimensional Hawkes processes, each of which contains I events. We represent the impact functions by D basis functions and set the maximum number of clusters to be K . In the worst case, the computational complexity per iteration of our learning algorithm is $\mathcal{O}(KDN I^3 C^2)$. Fortunately, according to the analysis of the identifiability, we know that our algorithm can achieve robust clustering results in the case having small number of events when the cluster centers are discriminative enough and the sizes of different clusters are relatively balanced. Therefore, in most of situations, I will not be very large and our algorithm will not be time-consuming. Moreover, we can accelerate our algorithm from several aspects. In the objective function (B.5) of the M-step, the prior distribution of tensor \mathbf{A} corresponds to a sparse regularizer, so the elements of \mathbf{A} become sparse with the increase of iteration. To the elements close to zero, we can ignore their computations in the learning phase to reduce the computational complexity. Specifically, if the number of nonzero elements in each \mathbf{A}^k is comparable to C , then the computational complexity of our algorithm will be $\mathcal{O}(KDN I^2 C)$. Besides the benefits from the sparsity of parameters, we can also accelerate our algorithm via some heuristic approximations. For example, the number of cluster can be truncated in the learning phase. An empirical method is ignoring the cluster whose N_k is smaller than a certain threshold N_{\min} after several iterations. As a result, the final runtime

of our learning algorithm will be $\mathcal{O}(K'DNI^2C)$, where $K' \ll K$ is the actual number of clusters. Additionally, the parallel computing techniques can also be applied to further reduce the runtime of our algorithm.

It should be noted that the computational complexity of our method is comparable to its competitors. Again, take the MMHP as an example. Following the ODE-based algorithm in [30], the learning algorithm of MMHP discretizes each impact function into L points and estimates them via finite element analysis. Therefore, its computational complexity per iteration is $\mathcal{O}(DNI(I^2C^2 + L(C + I)))$. Similarly, when the parameters of each Hawkes process is sparse, the computational complexity will reduce to $\mathcal{O}(DNI(IC + L(C + I)))$. According to the setting in [30, 22], generally $L \gg I$. Therefore, when K' is not very large, the computational complexity of our algorithm can be roughly compared to that of the MMHP.

4.4 Experiments

To demonstrate the feasibility and the efficiency of our **DMHP**-based clustering method, we compare it with the state-of-art methods, including the vector auto-regressive (**VAR**) method in [5], the Least-Squares (**LS**) method in [39], and the multi-task multi-dimensional Hawkes process (**MMHP**) in [22], on both synthetic and real-world data. These three competitors first learn the features of event sequences and then apply the Dirichlet process Gaussian mixture model (**DPGMM**) [106] to cluster the event sequences. The VAR method discretizes asynchronous event sequences to time series and learns their transition matrices as features. The LS method also discretizes event sequences but with higher resolution — it ensures that there is at most one event happening in each interval. For each event sequence, a nonparametric Hawkes process is learned via solving a Least-Squares problem, and the integration of impact functions over time (i.e., its infectivity matrix of event types³) is used as its feature. Similarly, we implement the MMHP model, learning infectivity matrices as

³The infectivity matrix of each sequence $\Phi = [\phi_{cc'}]$, where the element $\phi_{cc'} = \int_0^\infty \phi_{cc'}(t)dt$.

Table 4.1: The clustering purity on the synthetic data with sine-like impact functions.

| C | K' | VAR+DPGMM | LS+DPGMM | MMHP+DPGMM | DMHP |
|-----|------|-----------|----------|---------------|---------------|
| 2 | 2 | 0.8821 | 0.9937 | 0.9980 | 0.9960 |
| | 3 | 0.8523 | 0.9764 | 0.9832 | 0.9938 |
| | 4 | 0.8272 | 0.9528 | 0.9793 | 0.9916 |
| | 5 | 0.7785 | 0.9059 | 0.9701 | 0.9896 |
| 5 | 2 | 0.5235 | 0.5639 | 0.5917 | 0.9888 |
| | 3 | 0.3860 | 0.5278 | 0.5565 | 0.9650 |
| | 4 | 0.2894 | 0.4365 | 0.5112 | 0.9354 |
| | 5 | 0.2543 | 0.3980 | 0.4656 | 0.9015 |

Table 4.2: The clustering purity on the synthetic data with piecewise constant impact functions.

| C | K' | VAR+DPGMM | LS+DPGMM | MMHP+DPGMM | DMHP |
|-----|------|-----------|----------|------------|---------------|
| 2 | 2 | 0.8705 | 0.9203 | 0.9289 | 0.9637 |
| | 3 | 0.8259 | 0.9044 | 0.9080 | 0.9472 |
| | 4 | 0.7698 | 0.8827 | 0.8858 | 0.9220 |
| | 5 | 0.7138 | 0.8155 | 0.8366 | 0.8910 |
| 5 | 2 | 0.5222 | 0.5589 | 0.5913 | 0.8075 |
| | 3 | 0.3618 | 0.4402 | 0.4517 | 0.7675 |
| | 4 | 0.2901 | 0.3365 | 0.3876 | 0.7058 |
| | 5 | 0.2476 | 0.2980 | 0.3245 | 0.6738 |

the features of event sequences. In our DMHP method, the number of clusters are decided heuristically via the truncation method. For evaluating these algorithms comprehensively, we apply two measurements. For the synthetic data with the labels of clusters, we use *clustering purity* to evaluate various methods. For the real-world data, we visualize the centers of clusters and apply the cross-validation method in [107, 108] to measure the *consistency* of clustering results.

4.4.1 Synthetic Data

We generate two synthetic data sets with various clusters using sine-like impact functions and piecewise constant impact functions respectively. In each data set, the real number of classes, denoted as K' , is set from 2 to 5. Each cluster contains 400 event sequences, and each event sequence contains 50(= M_n) events and C (= 2 or 5) event types. The

exogenous base intensity of each cluster yields to C Rayleigh distributions, whose hyperparameters are sampled uniformly from $[0, 1]$. Each sine-like impact function in the k -th cluster is formulated as $\phi_{cc'}^k = b_{cc'}^k(1 - \cos(\omega_{cc'}^k(t - s_{cc'}^k)))$, where $t \in [0, 10]$ and the hyperparameters $\{b_{cc'}^k, \omega_{cc'}^k, s_{cc'}^k\}$ are generated randomly from $[\frac{\pi}{5}, \frac{2\pi}{5}]$. Each piecewise constant impact function in the k -th cluster is the truncation of the corresponding sine-like impact function (i.e., $2b_{cc'}^k \times \text{round}(\frac{\phi_{cc'}^k}{2b_{cc'}^k})$).

We estimate each impact function via the basis representation in (4.1). In our DMHP learning algorithm, we use Gaussian basis functions, and apply the basis selection method proposed in Chapter 2 to decide the bandwidth and the number of Gaussian basis functions. The maximum number of clusters K is set to be $2K'$ for all methods. After learning clustering results, the clustering purity is calculated as

$$Purity = \frac{1}{N} \sum_{k=1}^K \max_{j \in \{1, \dots, K'\}} |\mathcal{W}_k \cap \mathcal{C}_j|,$$

where \mathcal{W}_k is the learned index set of the event sequences belonging to the k -th cluster, and \mathcal{C}_j is the real index set of the event sequences belonging to the j -th class, and N is the total number of event sequences.

Tables 4.1 and 4.2 show the clustering purity for various methods on the two synthetic data. Compared with the three competitors, our DMHP obtains much better clustering purity consistently in most situations, especially when the dimension of Hawkes process becomes large. The VAR method simply treats asynchronous event sequences as time series, which causes the loss of temporal information. Both the LS and the MMHP can learn Hawkes process for each individual sequence, which might suffer to the over-fitting problem in the case having few observations. Moreover, all these methods decompose event sequence clustering problem into two subproblems: learning features and applying DPGMM. Such a strategy is very sensitive to the quality of feature. The potential problems mentioned above have negative influences on learned features, and then lead to unsatisfying clustering

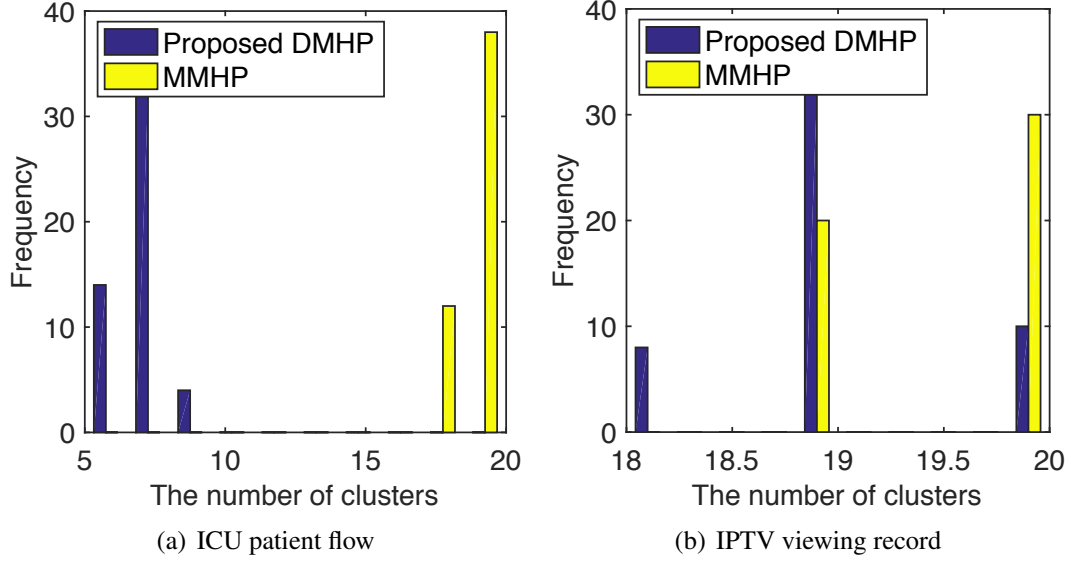


Figure 4.4: Histograms of clusters' numbers.

results. Our DMHP method, however, is a model-based clustering method, which learns clustering result directly and reduces the number of unknown variables greatly. As a result, our method avoids the problems of these competitors and obtains superior clustering results.

Moreover, the learning results of the synthetic data with piecewise constant impact functions prove that our DMHP method is relatively robust to the problem of model misspecification. Specifically, although our Gaussian basis cannot fit piecewise constant impact functions well, our method still outperforms other methods greatly on the clustering purity.

4.4.2 Real-world Data

We test our DMHP method on two real-world data sets. The first is the ICU patient flow data used in Chapter 6, which is extracted from the MIMIC II data set [109]. This data set contains the transition processes of 30308 patients among different kinds of care units. The patients can be clustered according to their transitions processes. The second data set is the IPTV data used in Chapters 2 and 3, which contains 7,100 IPTV users' viewing records

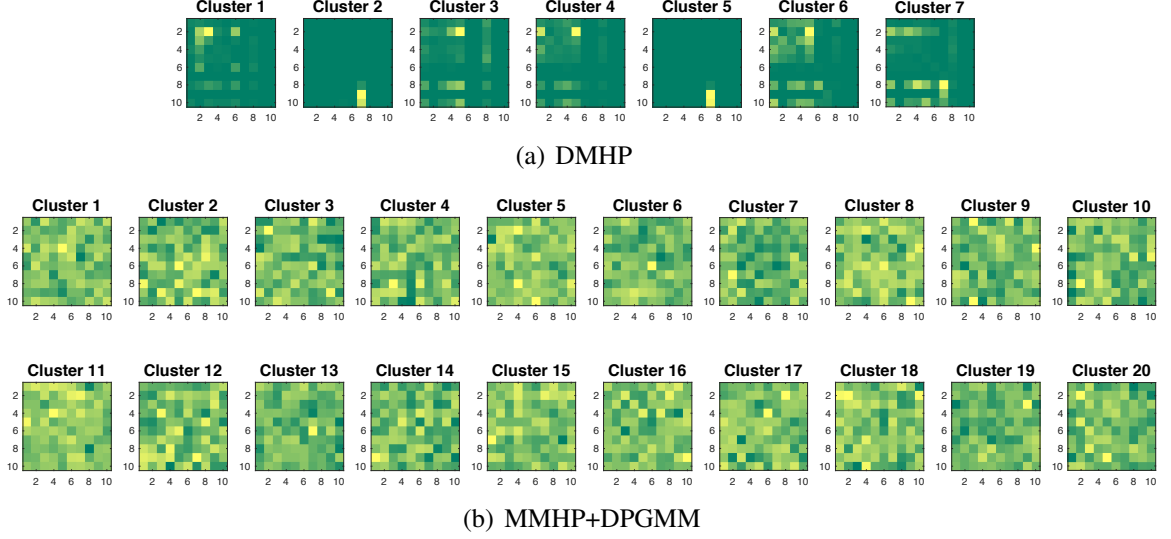


Figure 4.5: Comparison on infectivity matrices of ICU patient data.

collected via Shanghai Telecomm Inc. The TV programs are categorized into 16 classes and the viewing behaviors more than 20 minutes are recorded. The users can be clustered according to their viewing records.

According to the results of synthetic data, we compare our DMHP clustering method with its most powerful competitor — the MMHP+DPGMM method. For these two data sets, both these two methods set the maximum number of clusters to be 20. Applying the cross-validation method, we investigate the clustering consistency of these two methods. Specifically, for each data set, we test these two methods with 50 trials. In each trials, the data set is randomly divided into two folds. Taking one fold as a training set, we learn two models via the DMHP and the MMHP+DPGMM respectively, and then apply the models to cluster the remaining fold.

Because random sampling does not change the clustering structure of data, we can measure the clustering consistency in two aspects. The clustering method with high consistency should 1) have similar number of clusters in each trial, and 2) having similar pairwise relationships among different samples. Fig. 4.4 visualizes the histograms of the number of clusters for various methods on the two data sets. We can find that the MMHP+DPGMM method always over-segments data into too many clusters — the number of clusters reaches

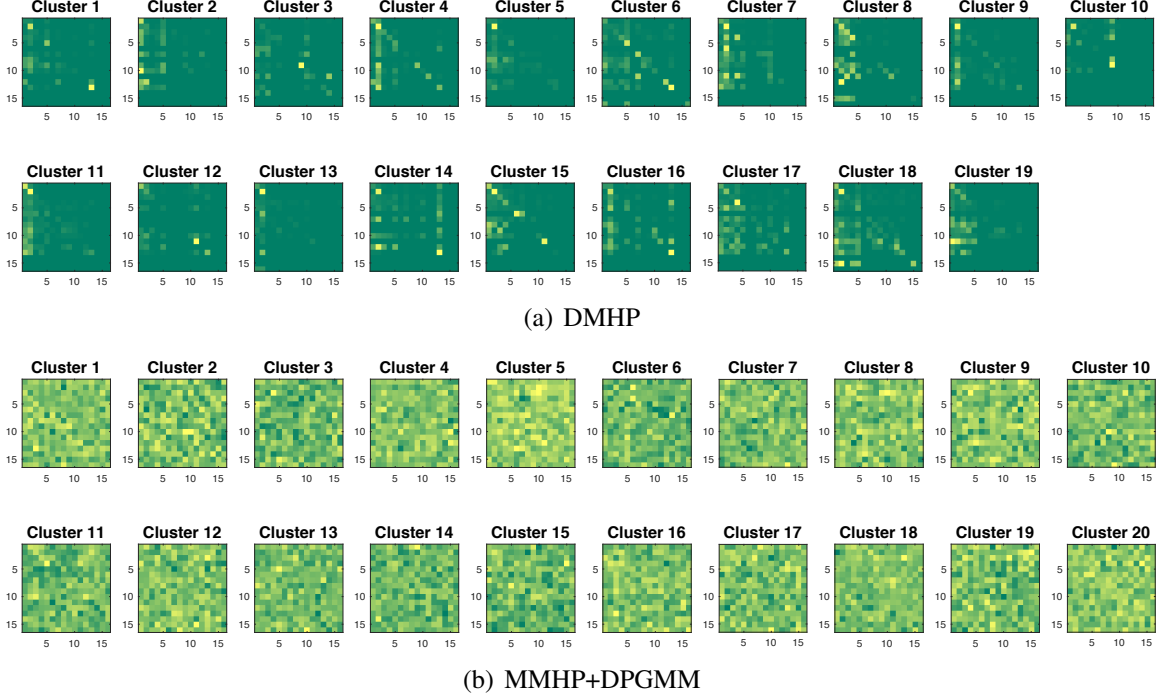


Figure 4.6: Comparison on infectivity matrices of IPTV data.

the maximum in most trials. Our DMHP method, however, can find more compact clustering structure. The distribution of the number of clusters concentrates to 7 and 19 for the two data sets, respectively. In our opinion, this phenomenon reflects the drawback of the feature-based method. The MMHP+DPGMM method first extracts features via learning the MMHP model and then learns clusters with the help of DPGMM. The clustering performance is highly dependent on the discriminative power of the features while the clustering structure is not considered sufficiently in the phase of extracting feature. A more direct comparison is given in Figs. 4.6 and 4.5: taking the infectivity matrices corresponding to different clusters⁴ as the representations of clustering results, we compare our DMHP method with the MMHP+DPGMM. The infectivity matrices obtained by our DMHP method are sparse and discriminative with each other, while those obtained by the MMHP+DPGMM are chaotic — although we consider the sparsity of each infectivity ma-

⁴Our DMHP calculates each cluster’s infectivity matrix directly from learned model, while the MMHP+DPGMM method calculates it via averaging infectivity matrices of the event sequences in the same cluster.

Table 4.3: The clustering consistency on real-world data sets.

| Method | VAR+DPGMM | LS+DPGMM | MMHP+DPGMM | DMHP |
|-------------|-----------|----------|------------|---------------|
| ICU Patient | 0.0896 | 0.1375 | 0.3239 | 0.3765 |
| IPTV User | 0.0438 | 0.0382 | 0.1351 | 0.1979 |

trix when learning the MMHP model, it cannot guarantee the average of the infectivity matrices within a cluster is still sparse and discriminative.

Besides the number of clusters, we further prove the superiority of our method through the pairwise relationships of samples. Specifically, taking the clustering result of the j -th trial as the reference, we enumerate all pairs of the event sequences within same cluster. In each of other trials, we count the number of the pairs still in same cluster, and calculate the minimum proportion of the pairs over all trials as the measurement of clustering consistency:

$$Consistency = \min_{j \in \{1, \dots, J\}} \sum_{j' \neq j} \sum_{(n, n') \in \mathcal{M}_j} \frac{1\{k_n^{j'} = k_{n'}^{j'}\}}{(J-1)|\mathcal{M}_j|},$$

where $J = 50$ is the number of trials. In the j -th trial, $\mathcal{M}_j = \{(n, n') | k_n^j = k_{n'}^j\}$ is the set of the pairs in the same cluster, and k_n^j is the index of cluster of the n -th event sequence. Table 7.2 shows the performance of various methods on the clustering consistency. We can find that our method outperforms other methods obviously, which means that the clustering result obtained via our method is more stable and consistent than other methods' results.

4.5 Summary

In this chapter, we propose a model-based clustering method for asynchronous event sequences, combining Dirichlet processes with Hawkes processes. The identifiability of the model is proven. A variational inference algorithm is designed to learn the mixture model, which has good performance on the convergence and the computational complexity. We demonstrate the feasibility and the effectiveness of our work on both synthetic and real-

world data. In the future, we plan to further improve our learning algorithm with optimal allocation strategies leveraging reinforcement learning and extend our work to more applications.

CHAPTER 5

LEARNING HAWKES PROCESSES FROM SHORT DOUBLY-CENSORED SEQUENCES

5.1 Introduction

Despite the usefulness of Hawkes processes, learning Hawkes processes often requires us to collect a large amount of event sequences with events occurring over a long observation window, which might be unavailable in many important practical applications. In particular, the observation window is likely to be very short and sequence-specific. Within an imagined universal window, each sequence is only observed with a corresponding short sub-interval of it, and the events outside this sub-interval are not observed — we call them short doubly-censored (SDC) event sequences. Existing learning algorithms of Hawkes processes, including ours in the previous three chapters, suffer to the over-fitting problem when they are directly applied to the SDC event sequences. What is worse, because the triggering pattern between some historical events and current ones are lost in the SDC event sequences, the model learned from the SDC event sequences are often unreliable.

This problem is a thorny issue in several practical applications, especially in those having time-varying triggering patterns. For example, the disease network learned from patients' admission records should evolve with the increase of the patients' ages, as Fig. 5.1 shows. However, it is very hard to track and record the patients' diseases on a life-time scale. Instead, we can only obtain their several admissions (even only one admission) during one or two years, which are just SDC event sequences.

Currently, many efforts have been made to learn point processes from imperfect observations. In the aspect of learning algorithm, the global [61] and local [62] likelihood maximization methods are used to learn Poisson processes. The pseudo MLE in [63] and

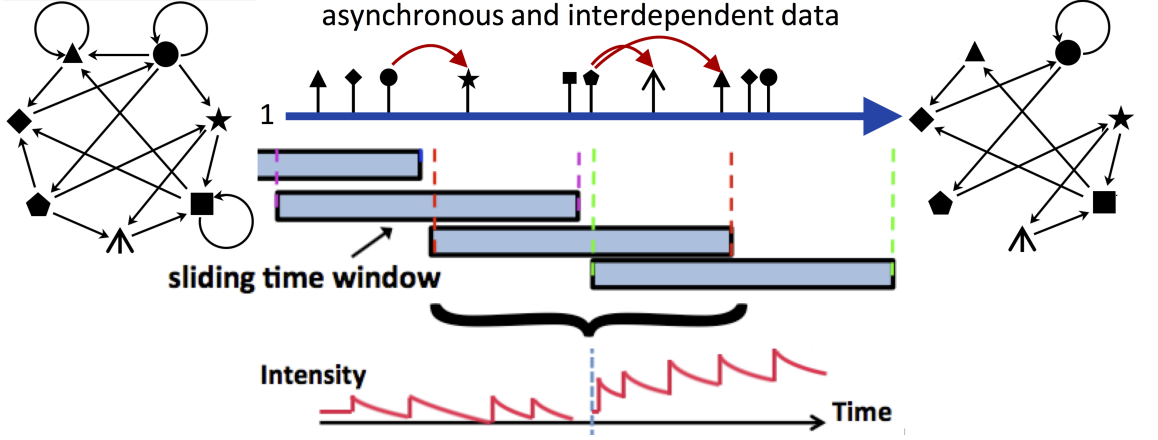


Figure 5.1: Illustration of a time-varying Hawkes processes, which can be used to model the dynamics of disease network over the age of patient.

the full MLE in [64] are applied to learn non-homogeneous Poisson processes. To learn Hawkes processes robustly, we introduce structural constraints, e.g., group-lasso, sparse, and low-rank regularizers in Chapters 2 and 3, to our learning algorithms. In the aspect of data, multiple imputation (MI) [65] is a general framework to stochastically impute incomplete or missing data from the current model in order to build a surrogate data set of observations. Recently, bootstrap methods have been introduced to the inference of point processes [72, 71, 73]. However, all these methods do not consider to learn Hawkes processes from SDC event sequences.

In this chapter, we propose a novel and simple data synthesis method to learn Hawkes processes having a longtime support from SDC event sequences. Specifically, given a set of SDC event sequences, we sample predecessors/successors for each event sequence from potential candidates and stitch them together as a synthetic training data. In the sampling step, the probabilistic distribution of the predecessors/ successors is estimated according to the similarities between current sequence and its candidates, and the similarity is defined based on the information of time stamps and (optional) features of event sequences. We analyze the rationality and the feasibility of our data synthesis method in details, and discuss the necessary condition of the method. Experimental results show that our data synthesis method indeed helps to improve the robustness of various learning algorithms of

Hawkes processes. Especially in the case of time-varying Hawkes processes, applying our method as preprocessing in the learning phase achieves much better results than learning directly from the SDC event sequences, which is very meaningful for many practical applications, e.g., constructing dynamical disease network and learning long-term infectivity among different IT companies.

5.2 Learning from SDC Event Sequences

Suppose that the original complete event sequences are in a long observation window. However, the observation window in practice might be segmented into several intervals $\{T_b^n, T_e^n\}_{n=1}^N$, and we can only observe K_n SDC sequences $\{s_k^n\}_{k=1}^{K_n}$ in the n -th interval, $n = 1, \dots, N$. Although we can still apply maximum likelihood estimator to learn Hawkes processes, i.e.,

$$\min_{\Theta \geq 0} - \sum_{n,k} \log \mathcal{L}(s_k^n; \Theta), \quad (5.1)$$

the SDC event sequences would lead to serious over-fitting problem because of the loss of information. Can we do better in such a situation? In this work, we propose a data synthesis method based on a sampling-stitching mechanism, which extends the SDC event sequences to longer ones and enhances the robustness of our learning algorithms.

5.2.1 Data Synthesis via Sampling-Stitching

Denote the k -th SDC event sequence in the n -th interval as s_k^n . Because its *predecessor* is unavailable, if we learn the parameters of our model via (5.1) directly, we actually impose a strong assumption on our data that there is no event happening before s_k^n (or previous events are too far away from s_k^n to have influences on s_k^n). Obviously, this assumption is questionable — it is likely that there are influential events happening before s_k^n . A more reasonable strategy is enumerating potential predecessors and maximizing the expected

log-likelihood over the whole observation window:

$$\min_{\Theta \geq 0} - \sum_{n,k} \mathbb{E}_{\mathbf{s} \sim \mathcal{H}_{T_b^n}^c} [\log \mathcal{L}([\mathbf{s}, \mathbf{s}_k^n]; \Theta)]. \quad (5.2)$$

Here, $\mathbb{E}_{x \sim \mathcal{D}}[f(x)]$ represents the expectation of the function $f(x)$ with the random variable x obeying a distribution \mathcal{D} . $\mathbf{s} \sim \mathcal{H}_{T_b^n}^c$ represents all possible sequences before T_b^n . $\mathcal{L}([\mathbf{s}, \mathbf{s}_k^n]; \Theta)$ is the likelihood of the stitched sequence $[\mathbf{s}, \mathbf{s}_k^n]$.

The stitched sequence $[\mathbf{s}, \mathbf{s}_k^n]$ can be generated via **sampling** the SDC sequence \mathbf{s} from previous 1st, ..., $(k-1)$ -th intervals and **stitching** it to \mathbf{s}_k^n . The sampling process obeys the probabilistic distribution of the stitched sequences. Given \mathbf{s}_k^n , we can compute its similarity between its potential predecessor $\mathbf{s}_{k'}^{n'}$ in $[T_b^{n'}, T_e^{n'}]$ as

$$w(\mathbf{s}_{k'}^{n'}, \mathbf{s}_k^n) = \begin{cases} F(T_b^n, T_e^{n'}) \underbrace{F(f_k^n, f_{k'}^{n'})}_{\text{optional}}, & T_e^{n'} \leq T_b^n, \\ 0, & \text{otherwise.} \end{cases} \quad (5.3)$$

Here, we use $F(a, b) = \exp(-\|b - a\|_2^2 / \sigma_s)$ as a predefined similarity function with parameter σ_s . f_k^n is the feature of \mathbf{s}_k^n . Note that the availability of the feature is optional — even if the feature of sequence is unavailable, we can still define the similarity measurement purely based on time stamps. The normalized $\{w(\mathbf{s}_{k'}^{n'}, \mathbf{s}_k^n)\}$ provides us with the probability that $\mathbf{s}_{k'}^{n'}$ appears before \mathbf{s}_k^n (i.e., $p(\mathbf{s}_{k'}^{n'} | \mathbf{s}_k^n) \propto w(\mathbf{s}_{k'}^{n'}, \mathbf{s}_k^n)$). Then, we can sample $\mathbf{s}_{k'}^{n'}$ according to the categorical distribution (i.e., $\mathbf{s}_{k'}^{n'} \sim \text{Category}(w(\cdot, \mathbf{s}_k^n))$). Fig. ?? illustrates the sampling-stitching process.

Obviously, such a sampling-stitching mechanism can be applied L times iteratively to the SDC sequences in both backward and forward directions. Algorithm 5 shows the scheme of our data synthesis method in details. As a result, we can synthesize long event sequences from original SDC event sequences. Specifically, we can represent a stitched event sequence as $\mathbf{s}_{stitch} = [\mathbf{s}_1, \dots, \mathbf{s}_{2L+1}]$, $\mathbf{s}_l \in \{\mathbf{s}_k^n\}$, $l = 1, \dots, 2L+1$, whose probability

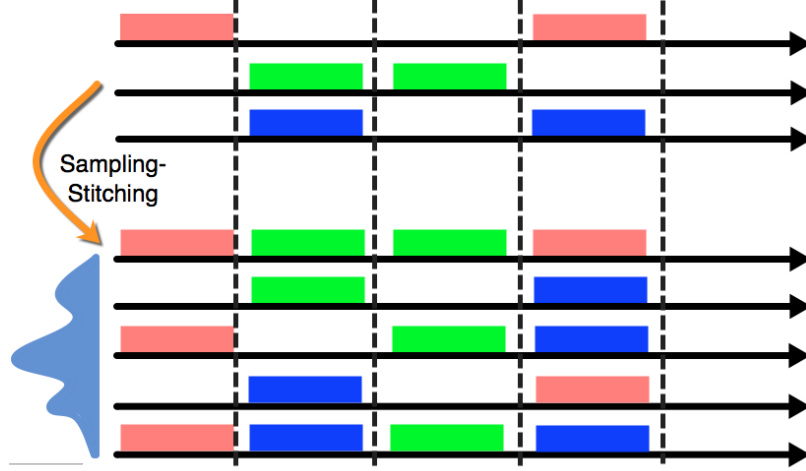


Figure 5.2: The illustration of our sampling-stitching process.

is

$$p(\mathbf{s}_{stitch}) \propto \prod_{l=1}^{2L} w(\mathbf{s}_l, \mathbf{s}_{l+1}). \quad (5.4)$$

5.2.2 Justification

After applying our data synthesis method, we obtain many stitched event sequences, which can be used as instances for estimating $\mathbb{E}_{\mathbf{s} \sim \mathcal{H}_{T_b}^c} [\log \mathcal{L}([\mathbf{s}, \mathbf{s}_k^n]; \Theta)]$. Specifically, taking advantage of the stitched sequences, we can rewrite the learning problem in (5.2) approximately as

$$\min_{\Theta \geq 0} - \sum_{\mathbf{s}_{stitch} \in \mathcal{S}} p(\mathbf{s}_{stitch}) \log \mathcal{L}(\mathbf{s}_{stitch}; \Theta). \quad (5.5)$$

We can find that (5.2) actually is the minimum cross-entropy estimation. The $p(\mathbf{s}_{stitch})$ represents the “true” probability of the stitched sequence, which is estimated based on the predefined similarity measurement. The likelihood $\mathcal{L}(\mathbf{s}_{stitch}; \Theta)$ represents the “unnatural” probability that the stitched sequence happens, which is estimated based on the definition in (1.4). Our data synthesis method takes advantage of the information of time stamps and (optional) features, and guarantees the rationality of the $p(\mathbf{s}_{stitch})$. For example, the

Algorithm 5 Data Synthesis via Sampling-Stitching

```
1: Input: SDC sequences  $\{s_k^n\}$ . The number of iterations  $L$ . The number of trials  $U$ .  
   Predefined parameter  $\sigma_s$   
2: Output: Stitched sequences  $\mathcal{S}$ .  
3: Initialize  $\mathcal{S} = \emptyset$ .  
4: Compute similarity measurement  $\{w\}$  via (5.3).  
5: For  $u = 1 : U$   
6:   For each  $s_k^n$   
7:      $s_{stitch} = s_k^n, s_b = s_e = s_k^n, l_b = l_e = 1$ .  
8:     Backward:  
9:     While  $l_b \leq \min\{n - 1, L\}$   
10:      Sample  $s_{k'}^{n'} \sim \text{Category}(w(\cdot, s_b))$ .  
11:       $s_b = s_{k'}^{n'}, s_{stitch} = [s_b, s_{stitch}]$ .  
12:       $l_b = n'$ .  
13:     End  
14:     Forward:  
15:     While  $l_e \leq \min\{N - n, L\}$   
16:      Sample  $s_{k'}^{n'} \sim \text{Category}(w(s_e, \cdot))$ .  
17:       $s_e = s_{k'}^{n'}, s_{stitch} = [s_{stitch}, s_e]$ .  
18:       $l_e = n'$ .  
19:     End  
20:      $\mathcal{S} = \mathcal{S} \cup s_{stitch}$ .  
21:   End  
22: End
```

likelihood of a sequence generally reduces with the increase of observation time window. The proposed probability $p(s_{stitch})$ yields to the same pattern — according to (5.4), the longer a stitched sequence is, the smaller its probability becomes.

The set of all possible stitched sequences (i.e., the \mathcal{S} in (5.5)) is very large, whose cardinality is denoted as $|\mathcal{S}| = \mathcal{O}(\prod_{n=1}^N K_n)$. In practice, we cannot and do not need to enumerate all possible combinations. An empirical setting is making the number of stitched sequences comparable to that of original SDC event sequences, i.e., generating $\mathcal{O}(\sum_{n=1}^N K_n)$ stitched sequences. In the following experiments, we just apply 5(= U) trials and generate 5 stitched sequences for each original SDC event sequence, which achieves a trade-off between the computational complexity of our method and its performance.

5.2.3 Feasibility

It should be noted that our data synthesis method is only suitable for those complicated point processes whose historical events have influences on current and future ones. Specifically, we analyze the feasibility of our method for two typical point processes.

Poisson Processes. Theoretically, our data synthesis method cannot improve learning results if the SDC event sequences are generated via Poisson processes. For Poisson processes, the happening rate of current and future events is independent of historical events. In other words, the intensity function of each interval can be learned independently based on the SDC event sequences. The stitched sequences do not provide us with any additional information.

Hawkes Processes. For Hawkes processes, whose intensity function is defined as (1.1), our data synthesis method can enhance the robustness of our learning algorithm. In particular, consider a long event sequence generated via a Hawkes process in the time window $[T_b, T_e]$. If we divide the time window into 2 intervals, i.e., $[T_b, T]$ and $(T, T_e]$, the intensity function corresponding to the second interval can be written as

$$\lambda_c(t) = \mu_c + \sum_{t_i \leq T} \phi_{cc_i}(t, t_i) + \sum_{T < t_i \leq T_e} \phi_{cc_i}(t, t_i), \quad t \in (T, T_e]. \quad (5.6)$$

When the events in the first interval are unobserved, we just have a SDC event sequence, and the second term in (5.6) is unavailable. Learning a Hawkes process directly from the SDC event sequence ignores the information of the second term, which has a negative influence on learning results. Our data synthesis method, however, leverages the information from other potential predecessors, and generates multiple long sequences with potentials. As a result, we can obtain multiple intensity functions sharing the second interval and calculate their log-likelihood functions, respectively. The weighted sum of these log-likelihood functions, as (5.5) does, provides us with an estimation of the log-likelihood of the real long event sequence.

Compared with learning from SDC event sequences directly, applying our data synthesis method can enhance the robustness of our learning algorithm in general, unless the term $\sum_{t_i \leq T} \phi_{cc_i}(t, t_i)$ is ignorable. Specifically, we can model the impact functions $\{\phi_{cc'}(t, s)\}_{c, c' \in \mathcal{C}}$ of Hawkes processes based on basis representation:

$$\phi_{cc'}(t, s) = \underbrace{\psi_{cc'}(t)}_{\text{Infectivity}} \times \underbrace{g(t-s)}_{\text{Triggering kernel}} = \sum_{m=1}^M a_{cc'm} \kappa_m(t) g(t-s). \quad (5.7)$$

Here, we relax the time-invariant assumption and decompose the impact functions into two parts:

- 1) *Infectivity*. $\psi_{cc'}(t) = \sum_{m=1}^M a_{cc'm} \kappa_m(t)$, $t \in [0, T]$, represents the infectivity of event type c' on event type c at time t .¹
- 2) *Triggering kernel*. $g(t) = \exp(-\beta t)$ measures the time decay of infectivity, where $t \in [0, T]$. It means that the infectivity of a historical event to current one reduces exponentially with the increase of temporal distance between them.

When β is very large, $\phi_{cc'}(t, s)$ decays rapidly with the increase of $t - s$, and the events happening long time ago can be ignored. In such a situation, our data synthesis method may not be able to improve learning results.

5.3 Implementation for Time-varying Hawkes Processes

Time-varying Hawkes process is a kind of physically-interpretable model for many natural and social phenomena. The proposed model in (5.7) reflects many common properties of real-world event sequences. First, the infectivity among various event types often changes smoothly in practice: in social networks, the interaction between two users changes smoothly, which is not established or blocked suddenly; in disease networks, the

¹When $M = 1$ and $\kappa_m(t) \equiv 1$, we obtain the simplest time-invariant Hawkes process, whose impact functions are shift-invariant over time. Relaxing the shift-invariant assumption of the impact functions, i.e., applying multiple Gaussian basis functions, we obtain a flexible time-varying Hawkes process model.

infectivity among diseases should change smoothly with the increase of patient's age. Applying Gaussian basis representation guarantees the smoothness of infectivity function. Second, the triggering kernel measures the decay of infectivity over time. According to existing work, the decay of infectivity is exponential approximately, which has been verified in many real-world data [34, 43, 110]. To learn time-varying Hawkes processes from SDC event sequences, we combine our data synthesis method with an EM-based learning algorithm. Applying our data synthesis method, we obtain a set of stitched event sequences $\mathbf{S} = \{\mathbf{s}_n\}$ and their appearance probabilities $\{p_n\}$, where $\mathbf{s}_n = \{(t_i^n, c_i^n)_{i=1}^{I_n} | t_i^n \in [T_b^n, T_e^n], c_i^n \in \mathcal{C}\}$ and p_n is calculated based on (5.3). According to (5.5, 5.7), we can learn the target Hawkes process via

$$\min_{\boldsymbol{\mu} \geq \mathbf{0}, \mathbf{A} \geq \mathbf{0}} - \sum_{n=1}^{|\mathbf{S}|} p_n \log \mathcal{L}(\mathbf{s}_n; \boldsymbol{\Theta}) + \gamma \mathcal{R}(\mathbf{A}). \quad (5.8)$$

Here, $\boldsymbol{\Theta} = \{\boldsymbol{\mu} = [\mu_c], \mathbf{A} = [a_{cc'm}]\}$ represents the parameters of our model. The vector $\boldsymbol{\mu}$ and the tensor \mathbf{A} are nonnegative. Based on (1.4), the log-likelihood function is

$$\begin{aligned} \log \mathcal{L}(\mathbf{s}_n; \boldsymbol{\Theta}) &= \sum_{i=1}^{I_n} \log \lambda_{c_i}(t_i) - \sum_{c=1}^C \int_{T_b^n}^{T_e^n} \lambda_c(s) ds \\ &= \sum_{i=1}^{I_n} \log \left[\mu_{c_i^n} + \sum_{j < i} g(\tau_{ij}^n) \sum_{m=1}^M a_{c_i^n c_j^n m} \kappa_m(t_i^n) \right] \\ &\quad - \Delta^n \sum_{c=1}^C \mu_c - \sum_{c=1}^C \sum_{m=1}^M \sum_{i=1}^{I_n} \sum_{j \leq i} a_{cc_j^n m} G_{ij}, \end{aligned} \quad (5.9)$$

where $\tau_{ij}^n = t_i^n - t_j^n$, $G_{ij} = \int_{t_j^n}^{t_i^n} \kappa_m(s) g(s - t_j^n) ds$, and $\Delta^n = T_e^n - T_b^n$. $\mathcal{R}(\mathbf{A})$ represents the regularizer of parameters, whose weight is γ . Similar to the work in the previous chapters, we assume that the infectivity connections among different event types should be sparse. Accordingly, at each time t , the infectivity matrix $\boldsymbol{\Psi}(t) = [\psi_{cc'}(t)]$ should reflect the sparse structure. Therefore, here we impose the ℓ_1 -norm regularizer on the coefficient tensor \mathbf{A} , i.e., $\mathcal{R}(\mathbf{A}) = \|\mathbf{A}\|_1 = \sum_{c,c',m} |a_{cc'm}|$.

We solve the problem via an EM algorithm. Specifically, when sparse regularizer is applied, we take advantage of the ADMM method, introducing an auxiliary variable $\mathbf{Z} = [z_{cc'm}]$ and a dual variable $\mathbf{U} = [u_{cc'm}]$ for \mathbf{A} and rewriting the objective function in (B.5) as

$$-\sum_n p_n \log \mathcal{L}(\mathbf{s}_n; \boldsymbol{\Theta}) + \frac{\rho}{2} \|\mathbf{A} - \mathbf{Z}\|_F^2 + \rho \text{tr}(\mathbf{U}^\top (\mathbf{A} - \mathbf{Z})) + \gamma \|\mathbf{Z}\|_1.$$

Here ρ controls the weights of regularization terms, which increases with the number of EM iterations. $\text{tr}(\cdot)$ computes the trace of matrix. Then, we can update $\{\boldsymbol{\mu}, \mathbf{A}\}$, \mathbf{Z} , and \mathbf{U} alternatively.

Update $\boldsymbol{\mu}$ and \mathbf{A} : Given the parameters obtained in the k -th iteration, we first apply Jensen's inequality to $-\sum_n \log \mathcal{L}(\mathbf{s}_n; \boldsymbol{\Theta})$ and obtain a surrogate objective function for $\boldsymbol{\mu}$ and \mathbf{A} :

$$\begin{aligned} \mathcal{Q}(\boldsymbol{\Theta}; \boldsymbol{\Theta}^k, \mathbf{Z}^k, \mathbf{U}^k) = & -\sum_{n=1}^N p_n \left\{ \sum_{i=1}^{I_n} \left[\sum_{j < i}^M \sum_{m=1}^M q_{ijm} \log \frac{g(\tau_{ij}^n) a_{c_i^n c_j^n m} \kappa_m(t_i^n)}{q_{ijm}} + q_i \log \frac{\mu_{c_i^n}}{q_i} \right] \right. \\ & \left. - \sum_{c=1}^C \sum_{m=1}^M \sum_{i=1}^{I_n} \sum_{j \leq i} a_{cc_j^n m} G_{ij} - \Delta^n \sum_{c=1}^C \mu_c \right\} + \frac{\rho}{2} \|\mathbf{A} - \mathbf{Z}^k + \mathbf{U}^k\|_F^2, \end{aligned}$$

where $q_i = \frac{\mu_{c_i^n}^k}{\lambda_{c_i^n}^k(t_i^n)}$ and $q_{ijm} = \frac{g(\tau_{ij}^n) a_{c_i^n c_j^n m}^k \kappa_m(t_i^n)}{\lambda_{c_i^n}^k(t_i^n)}$, and $\lambda_{c_i^n}^k(t_i^n)$ is calculated based on $\boldsymbol{\mu}^k$ and \mathbf{A}^k . Then, we can update $\boldsymbol{\mu}$ and \mathbf{A} via solving $\frac{\partial \mathcal{Q}}{\partial \boldsymbol{\mu}} = \mathbf{0}$ and $\frac{\partial \mathcal{Q}}{\partial \mathbf{A}} = \mathbf{0}$. Both of these two equations have closed-form solution:

$$\mu_c^{k+1} = \frac{\sum_n p_n \sum_{c_i^n=c} q_i}{\sum_n p_n \Delta_n}, \quad a_{cc'm}^{k+1} = \frac{\sqrt{B^2 - 4\rho C} - B}{2\rho}, \quad (5.10)$$

where

$$\begin{aligned} B &= \rho(u_{cc'm}^k - z_{cc'm}^k) + \sum_n p_n \sum_{c_i^n=c} \sum_{c_j^n=c', j \leq i} G_{ij}, \\ C &= -\sum_n p_n \sum_{c_i^n=c} \sum_{c_j^n=c', j < i} q_{ijm}. \end{aligned}$$

Algorithm 6 Learning Algorithm of Hawkes Processes

- 1: **Input:** Event sequences \mathcal{S} . The threshold V . Predefined parameters β , σ_κ , and γ .
 - 2: **Output:** Parameters \mathbf{A} and $\boldsymbol{\mu}$.
 - 3: $k = 0$, $\rho = 1$, initialize \mathbf{A}^k and $\boldsymbol{\mu}^k$ randomly. $\mathbf{Z}^k = \mathbf{A}^k$, $\mathbf{U}^k = \mathbf{0}$.
 - 4: **repeat**
 - 5: Obtain \mathbf{A}^{k+1} and $\boldsymbol{\mu}^{k+1}$ via (5.10).
 - 6: Obtain \mathbf{Z}^{k+1} via (5.11).
 - 7: Obtain \mathbf{U}^{k+1} via (5.12).
 - 8: $k = k + 1$, $\rho = 1.5\rho$.
 - 9: **until** $\|\mathbf{A}^k - \mathbf{A}^{k-1}\|_F < V$
 - 10: $\mathbf{A} = \mathbf{A}^k$, $\boldsymbol{\mu} = \boldsymbol{\mu}^k$.
-

Update \mathbf{Z} : Given \mathbf{A}^{k+1} and \mathbf{U}^k , we can update \mathbf{Z} via solving the following optimization problem:

$$\min_{\mathbf{Z}} \gamma \|\mathbf{Z}\|_1 + \frac{\rho}{2} \|\mathbf{A}^{k+1} - \mathbf{Z} + \mathbf{U}^k\|_F^2.$$

Applying soft-thresholding method, we have

$$\mathbf{Z}^{k+1} = S_{\frac{\gamma}{\rho}}(\mathbf{A}^{k+1} + \mathbf{U}^k), \quad (5.11)$$

where $S_\eta(x) = \text{sign}(x) \min\{|x| - \eta, 0\}$ is the soft-thresholding function.

Update \mathbf{U} : Given \mathbf{A}^{k+1} and \mathbf{Z}^{k+1} , we can further update dual variable as

$$\mathbf{U}^{k+1} = \mathbf{U}^k + (\mathbf{A}^{k+1} - \mathbf{Z}^{k+1}). \quad (5.12)$$

In summary, Algorithm 6 shows the scheme of our learning algorithm. Note that the algorithm can be applied to SDC event sequences directly via ignoring p_n 's.

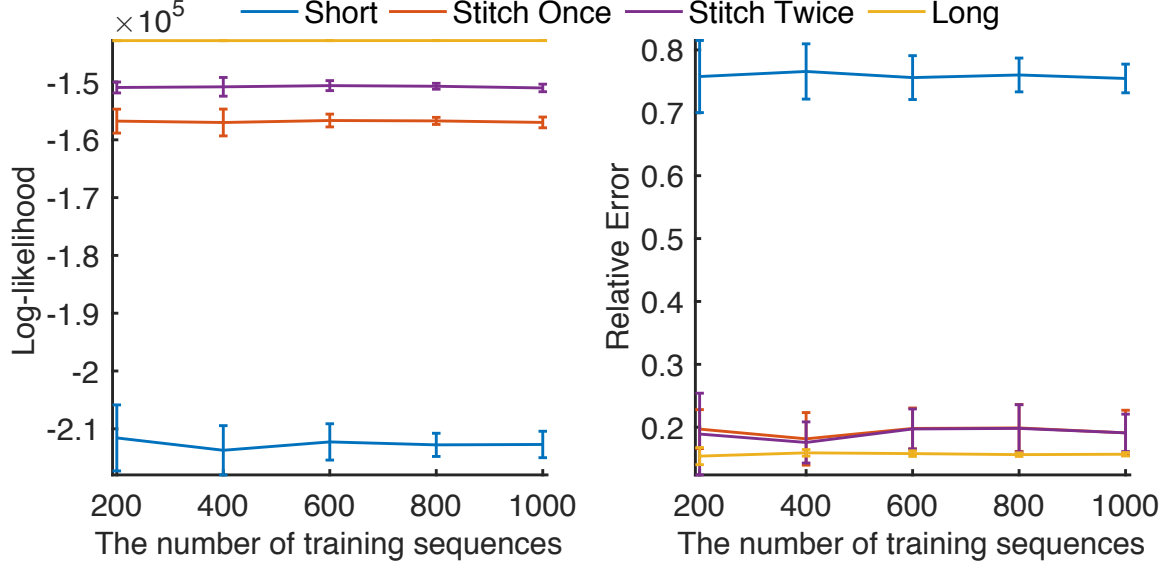
5.4 Experiments

5.4.1 Implementation Details

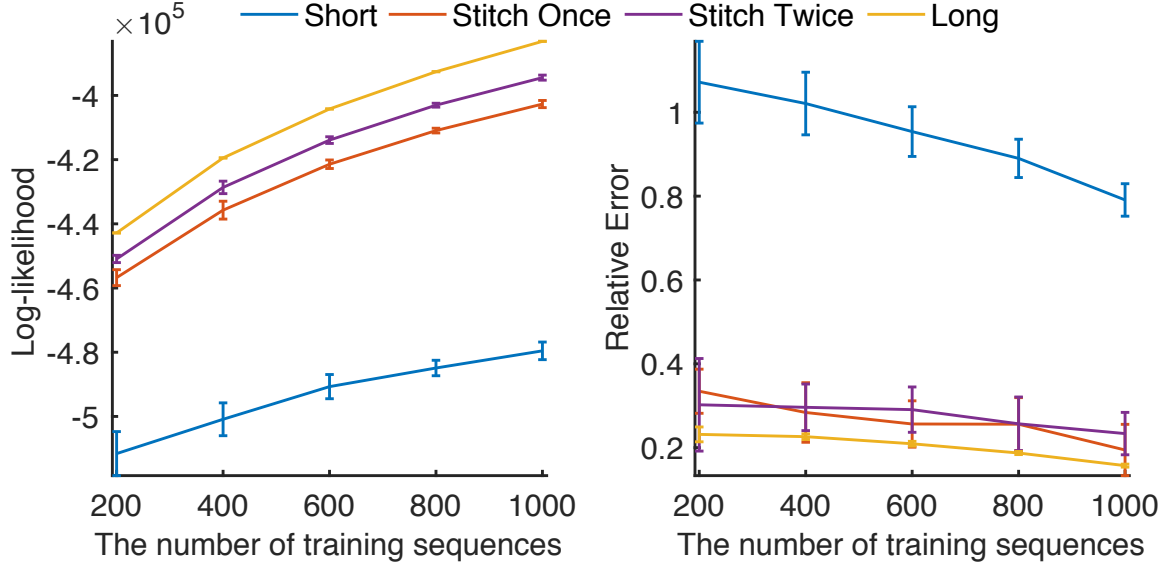
To demonstrate the usefulness of our data synthesis method, we combine it with various learning algorithms of Hawkes processes and learn different models accordingly from SDC event sequences. For time-invariant Hawkes processes, we consider two learning algorithms — our EM-based learning algorithm in Chapter 2 and the Least-Squares (LS) algorithm in [39]. For time-varying Hawkes processes, we apply our EM-based learning algorithm mentioned above. In the following experiments, we use Gaussian basis functions: $\kappa_m(t) = \exp(-(t - t_m)^2/\sigma_\kappa)$ with center t_m and bandwidth σ_κ . The number and the bandwidth of basis can be set according to the basis selection method proposed in Chapter 2. Additionally, we set $V = 10^{-4}$, $\gamma = 1$, and $\sigma_s = 1$ in our algorithm. We learn the Hawkes processes in two ways: 1) learning directly from SDC event sequences; 2) learning from stitched sequences generated via our data synthesis method.

5.4.2 Synthetic Data

The synthetic SDC event sequences are generated via the following method: 2000 complete event sequences are simulated in the time window $[0, 50]$ based on a 2-dimensional Hawkes process. The base intensity $\{\mu_c\}_{c=1}^2$ are generated randomly in the range $[0.1, 0.2]$. The parameter of triggering kernel, β , is set to be 0.2. For time-invariant Hawkes processes, we set the infectivity $\{\psi_{cc'}(t)\}$ to be 4 constants, which are generated randomly in the range $[0, 0.2]$. For time-varying Hawkes processes, we set $\psi_{cc'}(t) = 0.2 \cos(2\pi \frac{\omega_{cc'}}{50} t)$, where $\{\omega_{cc'}\}$ are random variables generated in the range $[1, 4]$. Given these complete event sequences, we select 1000 sequences as testing set while the remaining 1000 sequences as training set. To generate SDC event sequences, we segment the time window into 10 intervals, and just randomly preserve the events in one interval for each training sequences. To evaluate various methods, we test our data synthesis method in 10 trials, and



(a) Our learning algorithm



(b) Least squares algorithm

Figure 5.3: Comparisons on log-likelihood and relative error in the case of time-invariant Hawkes processes.

the relative error between real parameters Θ and estimation results $\hat{\Theta}$, i.e., $\frac{\|\Theta - \hat{\Theta}\|_2}{\|\Theta\|_2}$, and the log-likelihood of testing sequences are recorded.

Time-invariant Hawkes Processes. Fig. 5.3 shows the comparisons on log-likelihood and relative error for various methods. In Fig. 5.3(a) we can find that compared with the learning results based on complete event sequences, the results based on SDC event

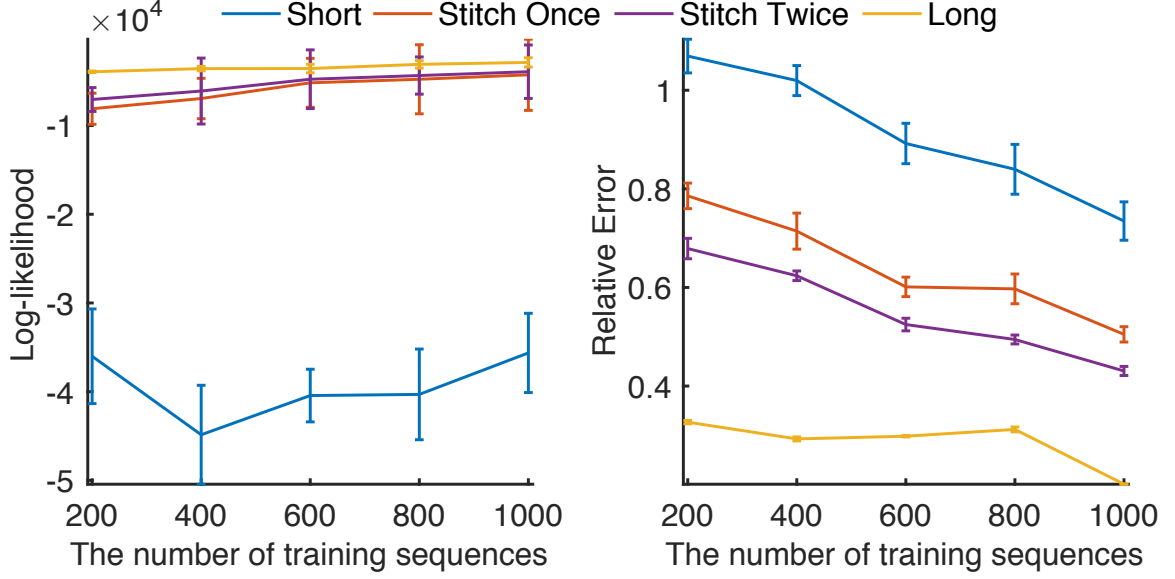


Figure 5.4: Comparisons on log-likelihood and relative error in the case of time-varying Hawkes processes.

sequences degrade a lot (i.e., lower log-likelihood and higher relative error) because of the loss of information. With the help of our data synthesis method, the learning results are improved consistently. Additionally, to demonstrate the universality of our method, we also apply our method to the LS algorithm [39]. In [39], the parameters of model are discrete points of shift-invariant impact functions $\{\phi_{cc'}(t)\}$. In Fig. 5.3(b) we can find that our method is also able to improve the learning results of LS algorithm in the case of SDC event sequences. Both the log-likelihood and the relative error obtained from stitched sequences approach to the results learned from complete sequences. It should be noted that although the contribution of the iterative framework of our method seems limited for time-invariant Hawkes processes (the relative error obtained via stitching SDC sequences twice is just comparable to that obtained via stitching once for both our learning algorithm and the LS algorithm), in the following experiments we can find that the iterative framework is more useful for time-varying Hawkes processes.

Time-varying Hawkes Processes. Fig. 5.4 shows the comparisons on log-likelihood and relative error for various methods. Similarly, the learning results are improved because of applying our method — higher log-likelihood and lower relative error are obtained and

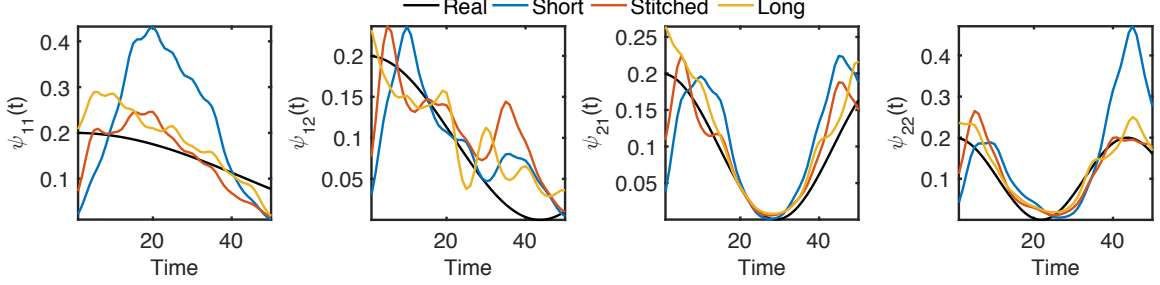


Figure 5.5: Comparisons on infectivity $\{\psi_{cc'}(t)\}$ in the case of time-varying Hawkes processes. The number of original SDC sequences is 200 and stitched via our method once.

their standard deviation (the error bars associated with curves) is shrunk. In this case, applying our method twice achieves better results than applying once, which verifies the usefulness of the iterative framework in Algorithm 5. Besides objective measurements, in Fig. 5.5 we visualize the infectivity functions $\{\psi_{cc'}(t)\}$. It is easy to find that the infectivity functions learned from stitched event sequences (red curves) are comparable to those learned from complete event sequences (yellow curves), which have small estimation errors of the ground truth (black curves).

5.4.3 Real-World Data

Besides synthetic data, we also test our method on real-world data, including the LinkedIn data collected via ourselves and the MIMIC III data set [111].

LinkedIn Data. The LinkedIn data set we collected online contains job hopping records of 3000 LinkedIn users in 82 IT companies. For each person, her/his time stamps corresponding to different companies are recorded as an asynchronous event sequence, and her/his profile (including education background, skill list, etc.) is treated as the feature associated with the event sequence. For each person, the attractiveness of a company is always time-varying. For example, a young man may be willing to join in startup companies and increase his income via jumping between different companies. With the increase of age, he would more like to stay in the same company and increase his income through internal promotion. In other words, the infectivity network among different companies should be

dynamical w.r.t. the age of employee. Unfortunately, the job records in the LinkedIn are short and doubly-censored — only the job hopping events in recent years are recorded. How to construct the dynamical infectivity network among different companies from SDC event sequences is still an open problem.

Applying our data synthesis method, we can stitch different users' job hopping sequences based on their ages (time stamps) and their profile (feature) and learn the dynamical network of company over time. In particular, we select 100 users with relatively complete job hopping history (i.e., the range of their working experience is over 25 years) as the testing set. The remaining 2900 users are selected as the training set. The log-likelihood of testing set in 10 trials is shown in Fig. 5.6(a). We can find that the log-likelihood obtained from stitched sequences is higher than that obtained from original SDC event sequences, and its standard deviation is bounded stably. Fig. 5.7(a) visualizes the adjacent matrix of infectivity network. The properties of the network verifies the rationality of our learning results: 1) the diagonal elements of the adjacent matrix are larger than other elements in general, which reflects the fact that most employees would like to stay in the same company and achieve a series of internal promotions; 2) with the increase of age, the infectivity network becomes sparse, which reflects the fact that users are more likely to try different companies in the early stages of their careers.

MIMIC III Data. The MIMIC III data set contains the admission records of over 40000 patients in critical care units of the Beth Israel Deaconess Medical Center between 2001 and 2012. For each patient, her/his admission time stamps and corresponding diagnoses of diseases (the ICD-9 code [112]) are recorded as an asynchronous event sequence, and her/his profile (including gender, race, and chronic history) is treated as the feature associated with the event sequence, which is represented as binary vector. As aforementioned, some work [110] has been done to extract time-invariant disease network from admission records, however, the real disease network should be time-varying w.r.t. the age of patient. Similar to the LinkedIn data, we only obtain SDC event sequences from the original

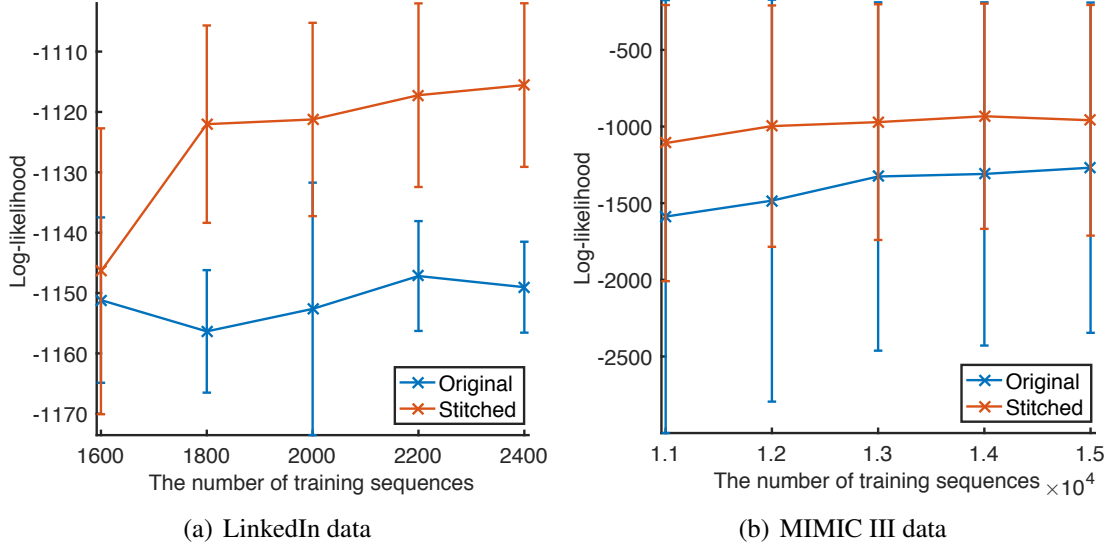


Figure 5.6: Comparisons on the log-likelihood of the testing data.

MIMIC III data set — the range of most admission records is just 1 or 2 years.

Applying our data synthesis method, we can leverage the information from different patients and stitch their event sequences based on their ages and their profile. Fig. 5.8 illustrates our method. Focusing on 600 common diseases in 12 categories, we select 15000 patients’ admission records randomly as the training set and 1000 patients with relatively complete records as the testing set. Fig. 5.6(b) shows that applying our data synthesis method indeed helps to improve the log-likelihood of the testing data. Furthermore, we visualize the adjacent matrix of dynamical network of disease categories in Fig. 5.7(b). We can find that: 1) with the increase of age the disease network becomes dense, which reflects the fact that the complications of diseases are more and more common when people become old; 2) the networks show that neoplasms and the diseases of circulatory, respiratory, and digestive systems have strong self-triggering patterns because the treatments of these diseases often include several phases and require patients to make multiple admissions; 3) for kids and teenagers, their disease networks (i.e., the “Age 0” and the “Age 10” networks) are very sparse, and their common diseases mainly include neoplasms and the diseases of circulatory, respiratory, and digestive systems; 4) for middle-aged people, the

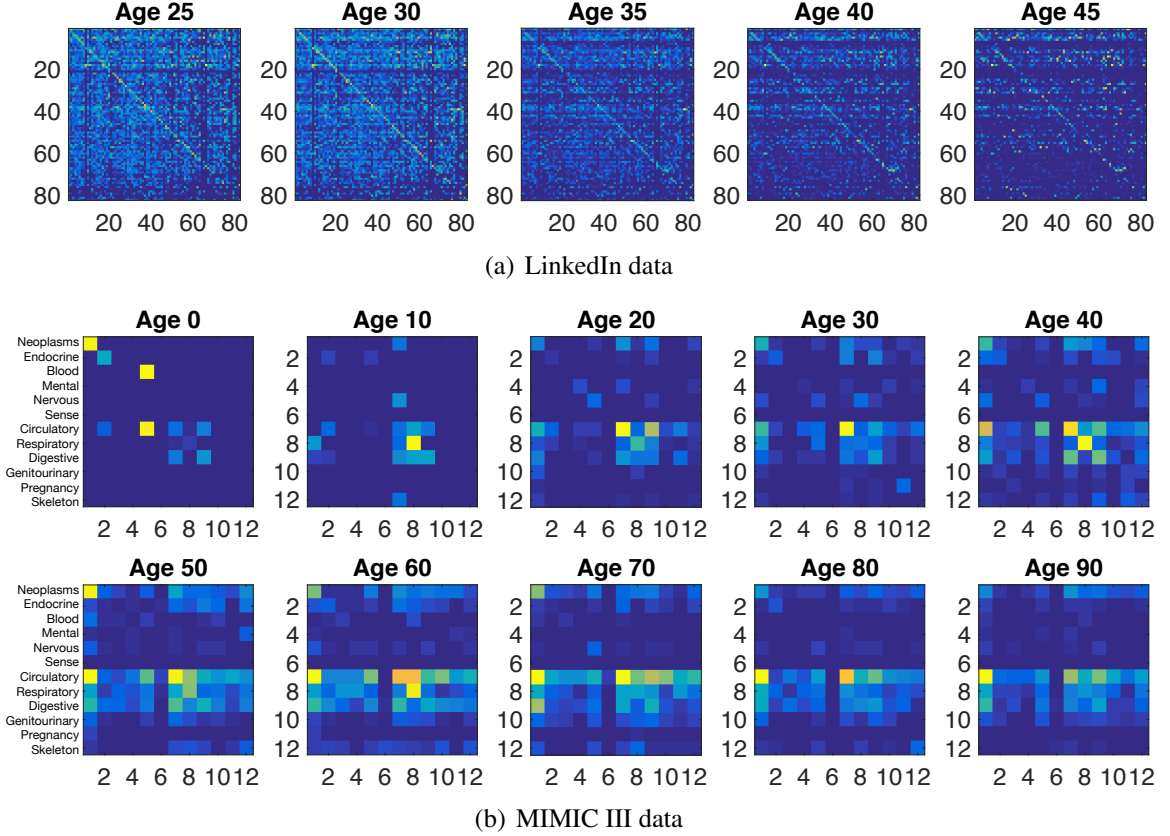


Figure 5.7: Comparisons on the infectivity $\{\psi_{cc'}(t)\}$ w.r.t. the age.



Figure 5.8: For each SDC sequence, i.e., incomplete disease history of a person in his lifetime, we design a mechanism to select other SDC sequences as predecessors/successors and synthesize a long sequence. Then, we can estimate the unobserved triggering patterns among diseases, i.e., the red dashed arrows, and construct a disease network.

reasons for their admissions are diverse and complicated so that their disease networks are dense and include many mutually-triggering patterns; 5) for longevity people, their disease networks (i.e., the “Age 80” and the “Age 90” networks) are relatively sparser than those of middle-aged people, because their admissions are generally caused by elderly chronic diseases.

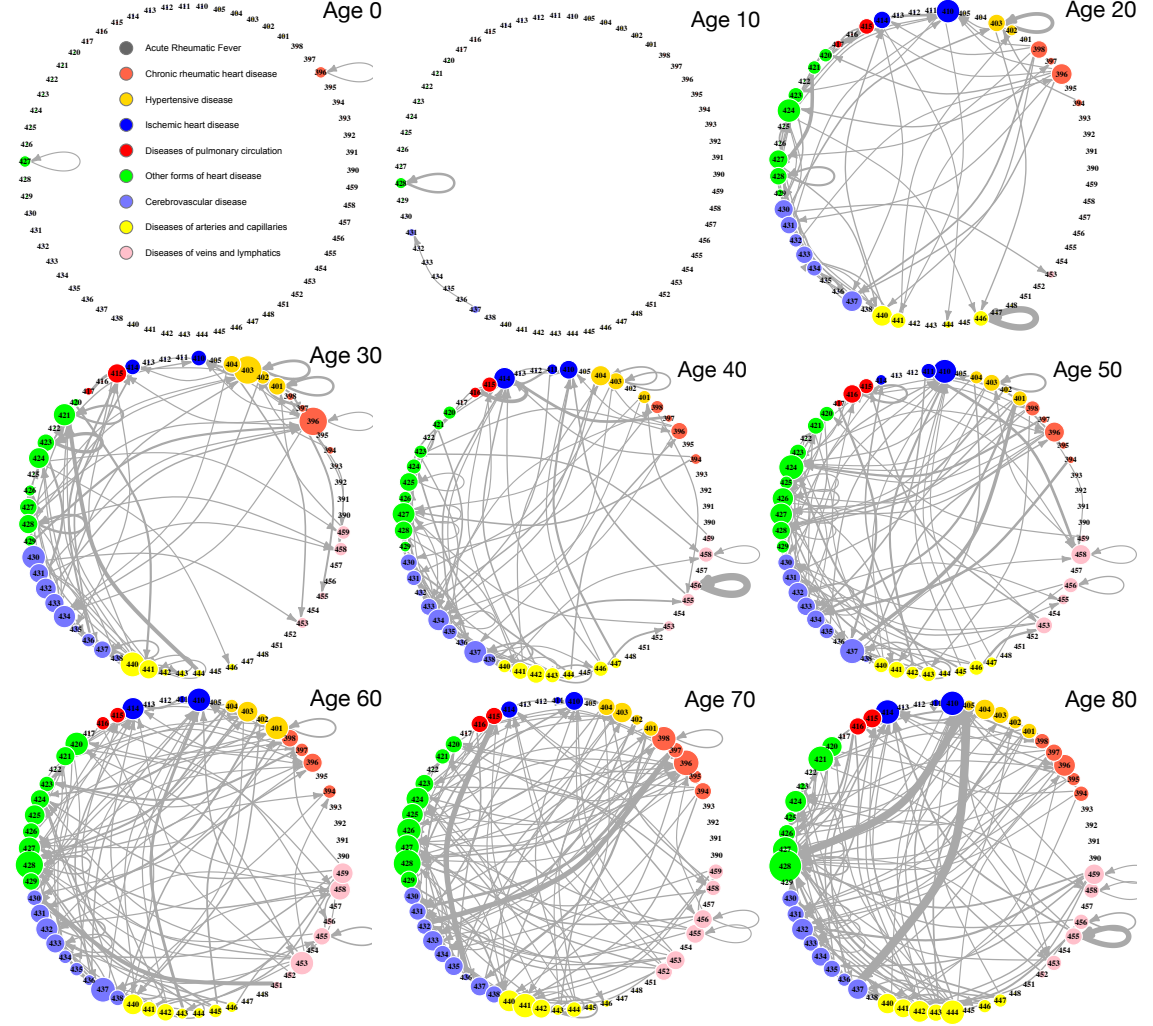


Figure 5.9: The network of circulatory system diseases in different ages. The diseases (nodes) are labeled with ICD-9 codes. The diseases belonging to different sub-categories are labeled with different colors. The size of the c -th node is $\sum_{c'} \psi_{cc'}(t)$, which reflects the importance of the c -th disease. The width of directed edge is set as $\psi_{cc'}(t)$, which indicates the strength of triggering pattern.

Additionally, we visualize the dynamical networks of the diseases of circulatory systems in Fig. 5.9, and find some interesting triggering patterns. For kids (the “Age 0” network), the typical circulatory diseases are “diseases of mitral and aortic valves” (ICD-9 396) and “cardiac dysrhythmias” (ICD-9 427), which are common for premature babies and the kids having congenital heart disease. For the old (the “Age 80” network), the network becomes dense. We can find that 1) as a main cause of death, “heart failure” (ICD-9 428) is triggered via multiple other diseases, especially “secondary hypertension”

(ICD-9 405); “secondary hypertension” is also likely to cause “other and ill-defined cerebrovascular disease” (ICD-9 437); 3) “Hemorrhoids” (ICD-9 455), as a common disease with strong self-triggering pattern, will cause frequent admissions of patients. In summary, the analysis above verifies the rationality of our result — the dynamical disease networks we learned indeed reflect the properties of human’s health trajectory. The list of ICD-9 codes can be found in https://en.wikipedia.org/wiki/List_of_ICD-9_codes_390-459:_diseases_of_the_circulatory_system.

5.5 Summary

In this chapter, we propose a novel data synthesis method to learn Hawkes processes from SDC event sequences. With the help of temporal information and optional features, we measure the similarities among different SDC event sequences and estimate the distribution of the long event sequences with potentials. Applying a sampling-stitching mechanism, we can successfully synthesize a large amount of synthetic event sequences and learn point process models robustly. Experimental results show that our data synthesis method improves the robustness of various models and learning algorithms.

CHAPTER 6

DISCRIMINATIVE LEARNING OF POINT PROCESSES AND PATIENT FLOW PREDICTION

6.1 Introduction

6.1.1 Motivations

In the previous chapter, we enhance the robustness of point processes' learning algorithms via introducing a data synthesis method as the pre-processing of imperfect data. In this chapter, we further think about the robustness problem in the viewpoint of model and algorithm. In particular, even if the data is sufficient, the learning algorithm might still fail when the assumption of the model disobeys the nature of the data. As a result, the model misspecification problem leads to bad learning results. Although in the previous chapters we have made some efforts to suppress the negative influence caused via model misspecification, e.g., taking advantage of basis representation for impact functions, relaxing the time-invariant assumption, etc, we need more flexible models to deal with challenging cases. Additionally, all learning algorithms designed in the previous chapters are generative, which maximize the likelihood of observed event sequences. However, in practice the data might not contain enough information for us to learn a complicated model in a generative way. In such a situation, a discriminative learning algorithm might be more suitable.

Based on the analysis above, we propose a discriminative learning method for point processes, which has potentials to solve the problems above. The novelties of our method include two points:

- **Flexible intensity function.** We can define the intensity function of point process via a more flexible way. Specifically, the intensity function can be constructed as a

composition of multiple linear and nonlinear functions, especially when each event is associated with a high-dimensional feature.

- **Discriminative learning algorithm.** Traditional generative learning method learns the joint distribution of events in the continuous time domain and the parameters of point processes are learned via the maximum likelihood estimation. In this work, however, we focus on learning the conditional probability of future events given historical events and target time stamps. As a result, we learn the parameters of model via maximizing the conditional probability, which generally requires fewer observations.

The proposed learning method is an attempt to connect deep learning techniques with point processes. The learning framework can be implemented as a neural network. The flexible representation of intensity function is obtained via the hidden layers while the discriminative learning algorithm corresponds to the output layer.

6.1.2 A Typical Application: Patient Flow Prediction

To demonstrate the usefulness of our method, we apply our method to predict the transition processes of patients among different care units, which achieves encouraging prediction results. Recent reports have highlighted an increasing demand for care units in the United States due to an improved life expectancy and a larger aging population [113]. The so-called practice of “patient boarding” refers to temporarily keeping critically-ill patients in their existing hospital location, such as the emergency department or the post anesthesia unit, while awaiting available CU bed [114, 115, 116, 117], which may result in suboptimal care, and increase both length of stay (LOS) and hospital mortality [118, 119]. System-level management of medical resources becomes even more critical for large numbers of critically-ill patients in the case of disasters and pandemics [120].

Such an urgent requirement gives rise to an important problem of predicting the transition processes of patients, known as the “patient flow” [121, 122] (see Fig. 6.1(a)). The

patient flow includes patients' duration time within each care unit and transition probability among different units, and is determined by a number of factors including patient's underlying condition and clinical state, disease progression, and availability of care team and care resources.

With the advent of comprehensive electronic health records (EHRs) and real-time streaming analytics [123], these factors can be captured and utilized to jointly model flow of patients within many care units. In particular, we can extract characteristic features of physiology in clinical data, or called phenotypes [124], from EHRs [125, 126, 127], and the features can be further applied to other problems like constructing disease network [110] and modeling patient flow [128, 129, 128] that we care in this work. The early work in [130] models patient flows in the viewpoint of treatment processes and proves that the treatment clustering information helps to model patient flow in emergency departments indeed. Following this strategy, the information of patients' treatment types is used to estimate the crowdedness of emergency departments in [131]. However, the methods above mainly focus on modeling the flow of patients having a certain kind of diseases from discrete time series or aggregate data. None of them attempt to model general patient flow in the continuous time domain. Therefore, the problem we aim to address involves predicting patients' destination CUs and durations simultaneously based on their medical records and continuously-documented clinical status.

However, predicting patient flow is a difficult task due to a number of factors, e.g., the collection and the storage of a huge amount of data, the lack of a systematic approach to resource management, etc. In the viewpoint of machine learning, the main challenges include:

- **Time-sensitivity.** The prediction of patient flow is a time-sensitive learning task, which requires us to predict the destination care unit of a patient and the dwell time within that care at the same time.
- **Feature selection.** The patient flow can be viewed as a time-varying transition pro-

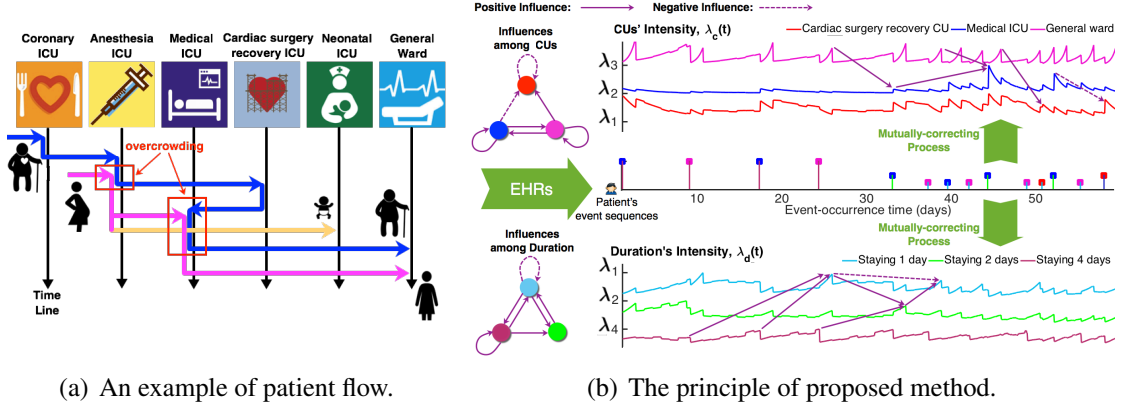


Figure 6.1: (a) The transition process of an old male patient having coronary heart disease may include the Coronary Care Unit for preoperative tests, the Anesthesia Services for cardiac surgery, the Cardiac Surgery Recovery unit, and finally the Medical ICU and general ward for nursing. The transition process of a pregnant woman having a premature baby may include the Anesthesia Services for a Caesarean section surgery, the Medical Care Unit for the mother, and the Neonatal Care Unit for the baby. There are overlaps between the elderly patient and the pregnant woman in the Anesthesia ICU and the Medical ICU. (b) The destination CUs and the duration days are represented via two event sequences. Along the time line, the color dots indicate various CUs and the color lines with various length indicate the durations (in units of days). Applying our mutually-correcting process model, the conditional intensity functions for CUs and durations are proposed to capture the positive and negative influences among unit types and durations, respectively.

cess in the continuous time domain, which is influenced by many medical factors, e.g., patients' health profiles, diagnoses, medications, nursing, etc. However, the relationships between these factors and the transition process are not fully explored and their importance for predicting patient flow is unknown.

- **Data sparsity and case imbalance.** Because most patients more often stay in general wards than transfer to other CUs, models and learning algorithms may suffer to sparse and imbalanced data — the general ward appears in most of transition processes while certain CUs may seldom appear.

Considering the challenges above, we need a predictive model that jointly captures the transitions and durations in patient flow. Moreover, the model should consider all influential factors and be robust to data sparsity and imbalance. To achieve this aim, we apply

our discriminative learning method of point processes, and propose a novel and efficient model that utilizes both time-invariant and time-varying features from patients' EHRs, to predict the patient flow. In particular, based on the unique characteristics of patient flow, we consider the transitions among the care units and the dwell time within each care unit as two separate events, which are jointly modeled via mutually-correcting processes, as depicted in Fig. 6.1(b). Based on the mutually-correcting process model, we learn the conditional distribution of transition and that of duration given historical events. We analyze the relationship between the conditional distribution and the conditional intensity function, showing that by using the proposed mutually-correcting process, we can formulate the learning problem as learning a multinomial logistic regression model that greatly simplifies the learning task.

Additionally, to overcome the data imbalance problem, we investigate several robust learning methods for imbalanced data and make comparisons for them. According to the comparison results, we applying a data synthesis method as the pre-processing of imbalanced data, which shows its superiority in our experiments: for the classes with extremely few samples, we synthesize some auxiliary samples from original ones to increase the number of training samples. Taking original samples and auxiliary ones as training samples, we can improve the robustness of our learning method greatly and obtain better performance in the testing phase.

6.2 Problem Statement and Data Analysis

6.2.1 Notations and Problem Statement

Suppose that we have N patients in a hospital having C CU departments. For each patient n , $n = 1, \dots, N$, her transition process among CUs is represented via an event sequence in continuous time, denoted as $\mathbf{s}_n = \{(c_i^n, d_i^n, t_i^n)\}_{i=1}^{I_n}$. Here, $t_i^n \in (0, T^n]$ is the time when a transition event happened, T^n is the length of observation time window, $c_i^n \in \mathcal{C}$, $\mathcal{C} = \{1, \dots, C\}$, is the destination CU of the transition, $d_i^n \in \mathcal{D}$, $\mathcal{D} = \{1, \dots, D\}$, is the

dwell time (measured by the number of duration days) of the patient in the previous CU (i.e., the c_{i-1}^n -th CU) before the transition, and N^u is the number of transitions.¹ The set of historical transitions before time t is denoted as $\mathcal{H}_t^n = \{(c_i^n, d_i^n, t_i^n) | t_i^n < t\}$.

Each event (c, d, t) , which means that a patient stays in a CU for d days before transferred to the c -th CU, is always accompanied by a series of medical services. According to the EHRs of patients, we classify various medical services into three categories: treatment, medication and nursing. The treatment contains M_{treat} items, including various medical tests, surgeries and therapies. The medication contains M_{med} items, including various medicines and their various usage methods. The nursing contains M_{nurse} items, including various nursing programs and records of patients' liquid inputs and outputs. We can extract binary feature vectors for patient n from her EHRs, denoted as $\mathbf{f}_i^n \in \{0, 1\}^{M_{treat}+M_{med}+M_{nurse}}$, $i = 1, \dots, N_u$. Here \mathbf{f}_i^n is a binary vector corresponding to the EHR of patient n when staying in the c_i^n -th CU, in which the elements corresponding to received services are 1's. It is the concatenation of three binary vectors corresponding to the three categories above. Besides the time-varying features mentioned above, a patient's EHR also contains M_p time-invariant features, including personal health profile like gender, age, chronic diseases, and diagnoses.² Similarly, we can extract a binary feature vector for the patient, denoted as $\mathbf{f}_0^n \in \{0, 1\}^{M_p}$.

For each patient n , given historical record $\mathcal{H}_{t_{i-1}^n}^n$ and features $\{\mathbf{f}_0^n, \mathbf{f}_1^n, \dots, \mathbf{f}_{i-1}^n\}$, we aim to predict the destination CU of the next transition (i.e., c_i^n) and the duration before the transition (i.e., d_i^n).

6.2.2 Data and Basic Statistics

We focus on the real-world data from MIMIC II database [132], from which 30685 patients staying in CUs are selected for training and testing. The CUs are categorized into $C = 8$

¹When $i = 1$, we do not consider the duration and set $d_i^n = \text{NULL}$.

²In our data set the diagnose is time-invariant because the patient flow for each patient is collected after a single diagnose.

departments, including the Coronary care unit (**CCU**), the Anesthesia care unit (**ACU**), the Fetal ICU (**FICU**), the Cardiac surgery recovery unit (**CSRU**), the Medical ICU (**MICU**), the Trauma Surgical ICU (**TSICU**), the Neonatal ICU (**NICU**), and the general ward (**GW**). According to the EHRs of the patients, the number of treatment items is $M_{treat} = 5627$, the number of medication items is $M_{med} = 405$, the number of nursing items is $M_{nurse} = 6808$, and the number of time-invariant features is $M_p = 4832$.

The data is representative, which reflects the following natures of patient flow. For each CU, the number of patients ever staying in it and the number of transitions directing to it are shown in Table 6.1. We can find that the data for various departments is imbalanced. On the one hand, most of the patients and transitions concentrate on certain CUs (i.e., GW, CCU, and CSRU) while few patients and transitions involve ACU and TSICU. The average duration days for each department is also listed. Except for NICU, the average dwell time of other department is within one week. Therefore, we categorize the duration times into $D = 8$ time intervals, include 1 day, 2 days, ..., 7 days and more than 1 week.

Interestingly we also observed that the transitions and the durations are weakly correlated with each other. Specifically, the correlation coefficient between the transition and the duration is about 0.2. The nature of the weak correlation between the transition and the duration is important for us to simplify our model, which will be shown in the following section.

Table 6.2 gives the proportions of nonzero elements in different feature domains w.r.t various CUs. Specifically, we count the number of nonzero elements in different feature domains for each CU and normalize the counts. The proportions reflect the importance of feature domains. We can find that patient's profile, treatment, and nursing are relatively important for all CUs, which contain most of nonzero features. On the contrary, the proportion of nonzero features from medication is relatively low. For TSICU and GW, most of nonzero features concentrate in the domain of treatment.

Table 6.1: Number of patients and transitions, and average durations (days) in each CU.

| Depts. | CCU | ACU | FICU | CSRU | MICU | TSICU | NICU | GW |
|------------|-------|------|-------|--------|-------|-------|-------|--------|
| # patients | 6,259 | 559 | 3,254 | 9,490 | 7,245 | 1,552 | 7,458 | 23,748 |
| # trans. | 7,030 | 631 | 3,525 | 10,679 | 8,903 | 1,628 | 7,657 | 28,118 |
| durations | 3.32 | 2.38 | 4.46 | 3.96 | 3.83 | 3.21 | 9.01 | 4.15 |

Table 6.2: The proportions of nonzero elements in different feature domains in each CU.

| Depts. | CCU | ACU | FICU | CSRU | MICU | TSICU | NICU | GW |
|------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Profile | 0.347 | 0.512 | 0.347 | 0.330 | 0.513 | 0.001 | 0.640 | 0.001 |
| Treatment | 0.505 | 0.354 | 0.505 | 0.562 | 0.342 | 0.995 | 0.241 | 0.996 |
| Nursing | 0.117 | 0.112 | 0.120 | 0.085 | 0.121 | 0.002 | 0.100 | 0.001 |
| Medication | 0.031 | 0.022 | 0.028 | 0.023 | 0.024 | 0.002 | 0.019 | 0.002 |

6.3 Proposed Method

We take advantage of the properties of patient flow and propose a mutually-correcting point process to describe the transitions among CUs and the durations in CUs respectively. The proposed model can be viewed as a specialization of a generalized parametric point process model. It has higher capability and can represent more complicated event sequences than existing popular point processes, e.g., the modulated Poisson process in [11], the Hawkes process in the previous chapters, and the self-correcting process in [26]. Our discriminative learning algorithm is applied, which corresponds to the multinomial logistic regression with the group-lasso regularizer. Both the feature selection problem and the imbalance of data are considered in our learning algorithm.

6.3.1 Mutually-Correcting Processes

As aforementioned, patient flow is a time-varying transition process in the continuous time domain. It generally has two important properties. Again, take the patient flow in Fig. 6.1(a) as an example:

- **High correlation between EHRs and patient flow.** A typical EHR consists of a patient’s profile (i.e., gender, age), her diagnose of certain diseases (i.e., ICD code),

and her treatment process (i.e., medications and nursing information). It reflects the patient's status and contains very useful information for predicting patient flow. Recall the previous cases shown in Fig. 6.1(a). For a man having coronary heart disease, the probability staying in the Coronary care unit is relatively high, while the probability staying in the Neonatal ICU is zero. On the contrary, for a premature baby, the probability staying in the Neonatal ICU is high while the probability staying in the Coronary care unit is very low. In more general cases, most of patients whose treatments involve surgeries are likely to have transitions among the Anesthesia care unit, the surgery recovery unit, and the general ward. These examples show that the patient flow is highly correlated with their EHRs.

- **Mutually-correcting across CUs.** Staying in the Coronary care unit is likely to increase the probability transferring to the Cardiac surgery recovery unit while suppress the probability transferring to the Neonatal ICU. It reflects that the duration of previous CU has a positive or negative influence on the transitions to following CUs, which is called mutually-correcting in our work.

Therefore, both the transitions among CUs and the durations in different CUs contain mutually-correcting patterns, which are highly dependent on EHR-based features. Additionally, taking the weak correlation between the transition and the duration into consideration, we propose a new point process model called mutually-correcting process to model the transitions and the durations respectively. Specifically, given the event sequence $\mathbf{s}_n = \{c_i^n, d_i^n, t_i^n\}_{i=1}^{I_n}$ of patient n , we decouple the event (c, d) into two independent events c and d , which correspond to two counting processes $\{N_c^n(t)\}_{c=1}^C$ and $\{N_d^n(t)\}_{d=1}^D$. Here $N_c^n(t)$ is the number of events that transferring patient n to the c -th CU after time t , while $N_d^n(t)$ is the number of events that staying in a CU d days after time t . We propose a generalized parametric model for the conditional intensity functions of these two counting

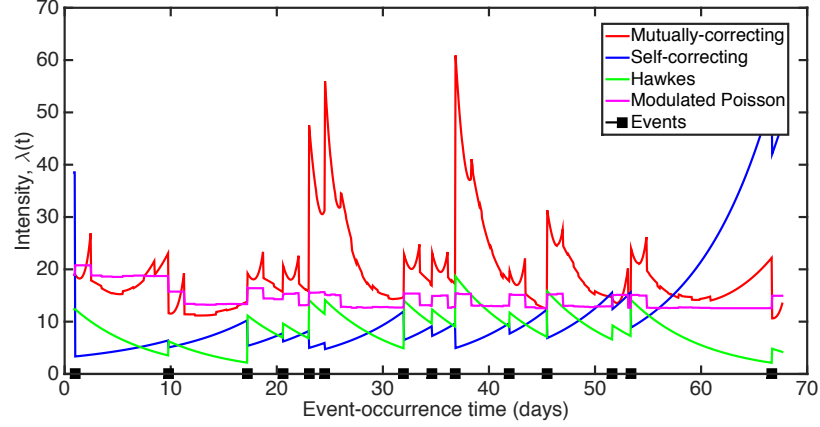


Figure 6.2: Comparison on conditional intensity function for various point processes. An event sequence is given and the conditional intensity functions of various point processes are shown.

Table 6.3: Comparison of various parametric point processes.

| Model | $f(x)$ | $g(t)$ | $h(t, t')$ | Constraints |
|------------------------------------|--------|-----------|----------------------------------|----------------------------|
| Modulated Poisson process | x | 1 | 1 | $\beta \leq 0 \leq \alpha$ |
| Hawkes process | x | 1 | $e^{-w(t-t')}$ | $\beta \leq 0 \leq \alpha$ |
| Self-correcting process | e^x | t | 1 | $\alpha, \beta \geq 0$ |
| Mutually-correcting process | e^x | $t - t_I$ | $e^{-\frac{(t-t')^2}{\sigma^2}}$ | — |

“—” means no constraints.

processes as follows:

$$\begin{aligned}
 \lambda_c^n(t) &= f(\alpha_c^\top \mathbf{f}_0^n g(t) - \beta_c^\top \sum_{i:t_i^n < t} \mathbf{f}_i^n h(t, t_i^n)), \\
 \lambda_d^n(t) &= f(\alpha_d^\top \mathbf{f}_0^n g(t) - \beta_d^\top \sum_{i:t_i^n < t} \mathbf{f}_i^n h(t, t_i^n)).
 \end{aligned} \tag{6.1}$$

$\lambda_c^n(t)$ represents the instantaneous rate of the event transferring patient n to the c -th CU at time t , while $\lambda_d^n(t)$ represents the instantaneous rate of the event staying in a CU d days. Here $\{\mathbf{f}_0^n, \mathbf{f}_i^n\}$ are time-invariant and time-varying features defined in Section 7.2. The term $\alpha^\top \mathbf{f}_0^n g(t)$ represents the temporal influence of time-invariant feature of the patient on event. The term $\beta^\top \sum_{i:t_i^n < t} \mathbf{f}_i^n$ represents the temporal influences of historical transitions \mathcal{H}_t^n on event. Here $f(\cdot)$, $g(\cdot)$ and $h(\cdot, \cdot)$ are predefined functions, which describes the increase or the decay of influences over time.

Actually, (7.5) provides a unified framework for many useful point processes, as Table 6.3 shows. In our mutually-correcting process model, we set $f(\cdot) = \exp(\cdot)$, $g(t) = t - t_I^n$, and $h(t, t') = \exp(-\frac{(t-t')^2}{\sigma^2})$, where t_I^n is the time stamp of the last event before time t for patient n . Our model extends traditional self-correcting process model [24] to multivariate case and further considers the temporal decay of influence from historical record. Compared with existing models, our model is more flexible. First, different from the self-correcting process, whose historical influence is time-invariant (i.e., $h(\cdot, \cdot) \equiv 1$), our model considers the time-varying historical influence as Hawkes process does. Second, the self-correcting process requires all parameters $\alpha = [\alpha_1, \dots, \alpha_C]$, $\beta = [\beta_1, \dots, \beta_C]$ to be nonnegative while the modulated Poisson and the Hawkes process require $\alpha \geq 0$ and $\beta \leq 0$. Our model, however, does not have such constraints. Such a relaxation increases the flexibility of our model. Fig. 6.2 compares the conditional intensity functions obtained via various 1-dimensional point processes. We can find that the conditional intensity function of the modulated Poisson process is piecewise constant. The conditional intensity always decreases for Hawkes process and increases for self-correcting process till new event comes. Our mutually-correcting process, however, is more flexible, which can capture both the increase and the decrease of intensity function between adjacent events.

Obviously, the conditional intensity function of our mutually-correcting process model can be rewritten as

$$\lambda_c^n(t) = \exp(\theta_c^\top \mathbf{f}_t^n), \quad \lambda_d^n(t) = \exp(\theta_d^\top \mathbf{f}_t^n). \quad (6.2)$$

$\mathbf{f}_t^n = [\mathbf{f}_0^{n\top}(t - t_I^n), (\sum_{t_i^n < t} \exp(-(t - t_i^n)^2/\sigma^2) \mathbf{f}_i^n)^\top]^\top \in \mathbb{R}^M$, $\theta_d = [\alpha_d^\top, \beta_d^\top]^\top$, $\theta_c = [\alpha_c^\top, \beta_c^\top]^\top$, $M = M_{treat} + M_{med} + M_{nurse} + M_p$. Such a simple representation inspires us to propose the following discriminative learning method with the help of multinomial logistic regression.

6.3.2 Discriminative Learning Algorithm

Traditional learning methods of point processes are generative, which aim to estimate the joint probability of all events via the maximum likelihood estimation. In the task of predicting patient flow, the optimization problem is

$$\max_{\Theta} \prod_{n,i} p(c_i^n, d_i^n, t_i^n | \mathcal{H}_{t_i^n}^n) (1 - P(T^n)),$$

where $p(c, d, t | \mathcal{H}_t^n)$ is the conditional probability of event (c, d) given historical record \mathcal{H}_t^n of patient n , and $P(T^n)$ is the cumulative probability transferring before T^n . The parameters of the model is represented as a matrix $\Theta = \{\theta_c, \theta_d\}_{c \in \mathcal{C}, d \in \mathcal{D}} \in \mathbb{R}^{M \times (C+D)}$. However, the generative learning methods may lack discrimination power because it aims to estimate the probability of the whole event sequence, rather than to classify or predict individual events given the historical record. The information of labels, e.g., the transition destination and the duration, is not fully used in the algorithm. Additionally, the generative learning methods will be at high risk of over-fitting when it comes to sparse and imbalanced data.

According to the analysis above, we apply our discriminative learning method in this work. Recall the problem we have: given current time t_{i-1}^n and historical record $\mathcal{H}_{t_{i-1}}^n$, we aim to maximize the probability that the patient n stay in a CU d_i^n days before being transferred to the c_i^n -th CU, i.e., $p(c_i^n, d_i^n | t_{i-1}^n, \mathcal{H}_{t_{i-1}}^n)$. Therefore, instead of estimating $p(c, d, t | \mathcal{H}_t^n)$ directly, we focus on the conditional probability of event (c, d) given current time t and the historical record, denoted as $p(c, d | t, \mathcal{H}_t^n)$. Note that this conditional probability is calculated as the normalized intensity function at time t . According to the weak correlation between the destination CUs and the duration days and the intensity functions in (7.5), we decouple the event (c, d) into two independent events c and d , and propose the

following cross-entropy-based loss function for our learning task.

$$\begin{aligned}
\mathcal{L}(\mathcal{S}; \Theta) &= - \sum_{n=1}^N \sum_{i=1}^{I_n} \left\{ \sum_{c=1}^C \mathbb{I}\{c_i^n = c\} \log p(c|t_{i-1}^n, \mathcal{H}_{t_{i-1}^n}^n) \right. \\
&\quad \left. + \sum_{d=1}^D \mathbb{I}\{d_i^n = d\} \log p(d|t_{i-1}^n, \mathcal{H}_{t_{i-1}^n}^n) \right\} \\
&= - \sum_{n=1}^N \sum_{i=1}^{I_n} \left\{ \log \left(\frac{\lambda_{c_i^n}^n(t_{i-1}^n)}{\sum_{c'} \lambda_{c'}^n(t_{i-1}^n)} \right) + \log \left(\frac{\lambda_{d_i^n}^n(t_{i-1}^n)}{\sum_{d'} \lambda_{d'}^n(t_{i-1}^n)} \right) \right\}.
\end{aligned} \tag{6.3}$$

Here $\mathbb{I}\{\text{statement}\}$ is an indicator of returning to 1 if the statement is truth, otherwise to 0.

Additionally, for exploring the relationship between the EHR-based feature and the patient flow, we consider the group sparsity of the parameter matrix of proposed model, denoted as $\|\Theta\|_{1,2}$. $\|\Theta\|_{1,2} = \sum_{m=1}^M \|\Theta_m\|_2$ sums the l_2 -norms of Θ 's rows Θ_m , $m = 1, \dots, M$. Here each dimension of feature is treated as a group. Introducing this term as a regularizer into the loss function, we achieve feature selection simultaneously when learning model — the rows corresponding to insignificant and noisy features will be suppressed to all zeros. Because the parameters of the model for predicting destination CUs and those for predicting durations are concatenated in Θ , the regularizer ensures that the useful features are shared via the two models. Such a feature selection strategy is also be used in [89, 133]. In summary, we learn our mutually-correcting process model via solving the following optimization problem:

$$\min_{\Theta} \mathcal{L}(\mathcal{S}; \Theta) + \gamma \|\Theta\|_{1,2}, \tag{6.4}$$

where $\gamma \geq 0$ is the weight controlling the significance of regularizer. Recalling the formula of conditional intensity function in (6.2), we can easily find that (6.4) corresponds to a problem combining multinomial logistic regression with group-lasso regularization [84]. From the viewpoint of Bayesian inference, the loss function $\mathcal{L}(\mathcal{S}; \Theta)$ corresponds to the negative log-likelihood function of Θ given a series of samples, and the group-lasso regu-

larizer imposes a structural prior distribution on Θ [134, 135] such that the prior probability $p(\Theta) \propto \exp(-\gamma \sum_{m=1}^M \|\Theta_m\|_2)$.

We apply the idea of alternating direction method of multipliers (ADMM) [136] to convert the optimization problem to several sub-problems that are easier to solve. Specifically, by introducing an auxiliary variable \mathbf{X} and a dual variable \mathbf{Y} , we obtain the augmented Lagrangian of (6.4) as follows:

$$\min_{\Theta} \mathcal{L}(\mathcal{S}; \Theta) + \gamma \|\mathbf{X}\|_{1,2} + \rho \text{tr}(\mathbf{Y}^\top (\Theta - \mathbf{X})) + \frac{\rho}{2} \|\Theta - \mathbf{X}\|_F^2, \quad (6.5)$$

where $\rho > 0$ is the penalty parameter. It mainly controls the convergence of ADMM algorithm [137]. $\text{tr}(\cdot)$ computes the trace of matrix. We solve it via optimizing the following sub-problems iteratively:

Update Θ : In the k -th iteration, we optimize the following problem:

$$\Theta^{(k+1)} = \arg \min_{\Theta} \mathcal{L}(\mathcal{S}; \Theta) + \frac{\rho}{2} \|\Theta - \mathbf{X}^{(k)} + \mathbf{Y}^{(k)}\|_F^2.$$

Applying gradient descent algorithm, we update Θ as

$$\Theta^{(k+1)} = \Theta^{(k)} - \beta \nabla \mathcal{L}|_{\Theta^{(k)}} - \beta \rho (\Theta^{(k)} - \mathbf{X}^{(k)} + \mathbf{Y}^{(k)}), \quad (6.6)$$

where parameter $\beta > 0$ is the learning rate for updating parameters. $\nabla \mathcal{L}|_{\Theta^{(k)}}$ is the gradient of loss function $\mathcal{L}(\mathcal{S}; \Theta^{(k)})$ given current parameters $\Theta^{(k)}$, which is computed as

$$\begin{aligned} \nabla \mathcal{L}|_{\theta_c^{(k)}} &= \sum_{n=1}^N \sum_{i=1}^{I_n} \left(\frac{\lambda_c^{n,(k)}(t_{i-1}^n)}{\sum_{c'} \lambda_{c'}^{n,(k)}(t_{i-1}^n)} - \mathbb{I}\{c_i^n = c\} \right) \mathbf{f}_{t_{i-1}}^n, \\ \nabla \mathcal{L}|_{\theta_d^{(k)}} &= \sum_{n=1}^N \sum_{i=1}^{I_n} \left(\frac{\lambda_d^{n,(k)}(t_{i-1}^n)}{\sum_{d'} \lambda_{d'}^{n,(k)}(t_{i-1}^n)} - \mathbb{I}\{d_i^n = d\} \right) \mathbf{f}_{t_{i-1}}^n. \end{aligned}$$

Here $\lambda_c^{n,(k)}(t)$ and $\lambda_d^{n,(k)}(t)$ are estimates of conditional intensity functions given current parameters.

Update \mathbf{X} : The optimization problem is a simple linear model with group-lasso penalty [84, 138, 139]:

$$\mathbf{X}^{(k+1)} = \arg \min_{\mathbf{X}} \frac{\rho}{2} \|\boldsymbol{\Theta}^{(k+1)} - \mathbf{X} + \mathbf{Y}^{(k)}\|_F^2 + \gamma \|\mathbf{X}\|_{1,2}.$$

Denote \mathbf{X}_m as the m -th row of \mathbf{X} . Its subgradient equations are

$$\rho(\mathbf{X}_m - (\boldsymbol{\Theta}_m^{(k+1)} + \mathbf{Y}_m^{(k)})) + \gamma \mathbf{s} = 0, \quad (6.7)$$

where $\mathbf{s} = \frac{\mathbf{X}_m}{\|\mathbf{X}_m^{(k)}\|_2}$ if $\mathbf{X}_m^{(k)} \neq \mathbf{0}$ and \mathbf{s} is a vector with $\|\mathbf{s}\|_2 < 1$ otherwise. The solution of (6.7) is

$$\hat{\mathbf{X}}_m = \left(1 + \frac{\rho}{\gamma \|\mathbf{X}_m^{(k)}\|_2}\right)^{-1} (\boldsymbol{\Theta}_m^{(k+1)} + \mathbf{Y}_m^{(k)}),$$

and then, $\mathbf{X}_m^{(k+1)}$ is updated via

$$\mathbf{X}_m^{(k+1)} = \begin{cases} \mathbf{0}, & \text{if } \|\hat{\mathbf{X}}_m - (\boldsymbol{\Theta}_m^{(k+1)} + \mathbf{Y}_m^{(k)})\|_2 \leq \frac{\gamma}{\rho}, \\ \hat{\mathbf{X}}_m, & \text{otherwise.} \end{cases} \quad (6.8)$$

Update \mathbf{Y} :

$$\mathbf{Y}^{(k+1)} = \mathbf{Y}^{(k)} + (\boldsymbol{\Theta}^{(k+1)} - \mathbf{X}^{(k+1)}). \quad (6.9)$$

Repeating the steps above until convergence, we learn the parameter matrix of the model, and obtain $p(c|t, \mathcal{H}_t^n)$ and $p(d|t, \mathcal{H}_t^n)$ jointly. In summary, we give the scheme of our learning algorithm in Algorithm 7.

Our model and algorithm can be viewed as a trade-off between learning joint probability $p(c, d|t, \mathcal{H}_t^n)$ directly and learning the probabilities of transition and duration ($p(c|t, \mathcal{H}_t^n)$ and $p(d|t, \mathcal{H}_t^n)$) independently. On one hand, learning $p(c, d|t, \mathcal{H}_t^n)$ requires $\mathcal{O}(CD)$ pa-

Algorithm 7 Discriminative Learning of Mutually-Correcting Processes (DMCP)

```
1: Input: Patient flow  $\{s_n\}_{n=1}^N$ , parameters  $\gamma, \rho, \beta$ , error bound  $\epsilon = 0.01$ .
2: Output:  $\Theta$ .
3: Initialize  $\Theta^{(0)}$  randomly,  $X^{(0)} = \Theta^{(0)}$ ,  $Y^{(0)} = \mathbf{0}$ , outer iteration number  $k = 0$ 
4: repeat
5:   Inner iteration number  $l = 0$ ,  $\Theta^{(k,l)} = \Theta^{(k)}$ .
6:   repeat
7:     Update  $\Theta^{(k,l+1)}$  via (6.6).
8:      $l = l + 1$ .
9:   until  $\frac{\|\Theta^{(k,l)} - \Theta^{(k,l-1)}\|_2}{\|\Theta^{(k,l)}\|_2} \leq \epsilon$ 
10:   $\Theta^{(k+1)} = \Theta^{(k,l)}$ .
11:  Update  $X^{(k+1)}$  via (6.8).
12:  Update  $Y^{(k+1)}$  via (6.9).
13:   $k = k + 1$ .
14: until  $\frac{\|\Theta^{(k)} - \Theta^{(k-1)}\|_2}{\|\Theta^{(k)}\|_2} \leq \epsilon$ .
15:  $\Theta = \Theta^{(k)}$ .
```

rameters, which might lead to the over-fitting result. Our model, however, merely requires $\mathcal{O}(C + D)$ parameters. On the other hand, although we relax the weak correlation between the transition and the duration to an independence assumption, we do not really learn $p(c|t, \mathcal{H}_t^n)$ and $p(d|t, \mathcal{H}_t^n)$ independently. With the help of the group-lasso in (6.4), their correlation is preserved to some degree — the group sparsity of parameters is shared via $p(c|t, \mathcal{H}_t^n)$ and $p(d|t, \mathcal{H}_t^n)$ and the parameters are updated simultaneously.

It should be noted that our discriminative algorithm is not only suitable for mutually-correcting processes. Actually, we can use conditional intensity functions from arbitrary point processes to compute the conditional probabilities and the loss function in (6.3).

6.3.3 Enhancing Robustness to Imbalanced Data

As aforementioned, the imbalance of the data has a remarkable impact on the overall performance of patient work flow prediction, leading to the poor performance of duration and transition prediction of classes with minority samples (i.e., in the following experiments, the prediction accuracy of destination CUs with only a few patients transferring to CUs like ACU, FICU, TSICU, is relatively lower than other CUs with more patients like CCU,

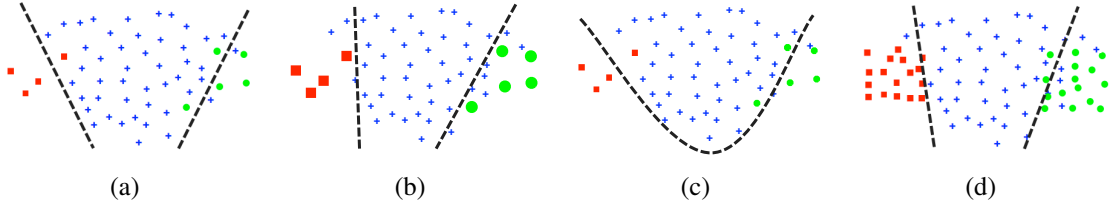


Figure 6.3: The simple 2-D example illustrating various methods to solve data imbalance problem: (a) The original data having 3 classes is shown, where blue crosses are samples of major class while red squares and green dots are samples of two minor classes, respectively. (b) The weighed data is shown, where the samples of minor classes have large weights (enlarged). (c) The hierarchical data is shown, where the samples are unchanged while a nonlinear binary classifier is learned. (d) The synthetic data is shown, where the minor classes are supplemented via auxiliary samples. The classifiers in the subfigures are shown as black dotted lines and curves.

SCRU, MICU, NICU). As the 2-D case in Fig. 6.3(a) shows, the classifier trained on imbalanced data will focus more on the classification accuracy of the class having sufficient samples while ignore the errors of the class having extremely few samples.

For suppressing the negative influence of data imbalance problem, several potential solutions are proposed and analyzed in depth.

- **Weighted data.** A possible way to increase the significance of the classes with few samples is adding the weights of the samples in the training phase [76, 77, 79, 78]. Specifically, we can rewrite the likelihood function in (6.3) as

$$-\sum_{n=1}^N \sum_{i=1}^{I_n} w_i \log \left(\frac{\lambda_{c_i^n}^n(t_{i-1}^n) \lambda_{c_i^n}^n(t_{i-1}^n)}{\sum_{c'} \lambda_{c'}^n(t_{i-1}^n) \sum_{d'} \lambda_{d'}^n(t_{i-1}^n)} \right),$$

where the weight w_i aims at suppressing the imbalance of the data. It should be large for the samples in the minor classes and small for those in the major ones (i.e., in our case, counting the number of labels $\{(c, d)\}$ in the training set, denoted as $\#\{(c, d)\}$, we calculate $w_i = \frac{1}{\log(1+\#\{(c, d)\})}$ if $c_i^n = c$ and $d_i^n = d$). Fig. 6.3(b) visualizes the weighed data, where the enlarged squares and dots are the samples with large weights. However, such a simple method might wrongly change the distribution of data and lead to wrong boundaries between classes. As a result, it is likely to increase

the classification accuracy of the classes with few training samples while decrease the classification accuracy of the classes with sufficient samples at the same time.

- **Hierarchical data.** Instead of learning one multi-class classifier directly with imbalanced data, we can rank classes according to the number of training samples and learn binary classifiers hierarchically [74, 75]. Specifically, in each step, we take the class with the largest number of training samples as “MAJORITY”, and the rest samples as a single class called “MINORITY”. Then, a binary classifier is trained on them and the samples of “MAJORITY” is removed from the training set. Repeating the steps above, we obtain a series of binary classifier from hierarchical data. The principle of this method is re-balancing data via merging minor classes. However, in practice, the merging step may lead the classes to be linear-inseparable, which increase the difficulty of training phase. In this case, as Fig. 6.3(c) shows, nonlinear binary classifier is required in each step, which relies on more complicated learning algorithms.
- **Synthetic data.** For overcoming the weaknesses of the two methods above, we propose a data synthesis method to solve the data imbalance problem: for the samples (feature vectors) in a minor class, we synthesize auxiliary samples for the class by sampling each element according to the distribution of corresponding elements of existing samples. Therefore, the auxiliary samples are similar but not identical to original ones. Supplementing these auxiliary samples to the minor classes as training samples, we obtain balanced training data. Our data synthesis method is actually based on an assumption that the dimensions of feature are independent with each other. Different from the two competitors mentioned above, which change the distribution of original data, our method can guarantee that the auxiliary samples obeys the same distribution with original data as long as the assumption is held. In the following experiments, we will show that applying our data synthesis method as the

pre-processing of training data, we can enhance the robustness of learning method and obtain superior testing results to its competitors.

6.3.4 Patient Flow Prediction

Given learned model Θ , we can predict patient flow for each patient n simply. Specifically, given historical record $\mathcal{H}_{t_{i-1}}$, we compute $p(c|t_{i-1}^n, \mathcal{H}_{t_{i-1}}^n)$ and $p(d|t_{i-1}^n, \mathcal{H}_{t_{i-1}}^n)$ for $c \in \mathcal{C}$ and $d \in \mathcal{D}$, respectively. The predicts of c_i^n and d_i^n are given as

$$\hat{c}_i^n = \arg \max_{c \in \mathcal{C}} p(c|t_{i-1}^n, \mathcal{H}_{t_{i-1}}^n), \quad \hat{d}_i^n = \arg \max_{d \in \mathcal{D}} p(d|t_{i-1}^n, \mathcal{H}_{t_{i-1}}^n). \quad (6.10)$$

6.3.5 The Connection with Deep Learning

Our discriminative point process learning method actually can be viewed as an implementation of a deep neural network. In particular, given the features associated with event sequences, the flexible intensity function actually provides us with a powerful representation of the features, which corresponds to multiple hidden layers of a forward neural network. The discriminative learning framework that maximizes the conditional probability of the events can be implemented as logistic regression, which corresponds to the output layer of the neural network. Such a deep learning-based discriminative learning framework is very suitable for dealing with the classification tasks of event sequences.

We can extend our discriminative learning method to more general situations. Suppose that we have a set of event sequences $\mathcal{S} = \{\mathbf{s}_n\}_{n=1}^N$, where $\mathbf{s}_n = \{(t_i^n, c_i^n, \mathbf{f}_i^n)\}_{i=1}^{I_n}$ is the n -th event sequence and \mathbf{f}_i^n is the high dimensional feature associated with the event (t_i^n, c_i^n) . We can represent intensity function as follows:

$$\lambda_c(t) = \underbrace{F^L(F^{L-1}(\dots(F^2(F^1(t, \mathcal{H}_t))\dots))}_{\text{composite function}}).$$

Here, $\mathcal{H}_t = \{(t_i, c_i, \mathbf{f}_i) | t_i < t\}$ contains all historical events and features before time t ,

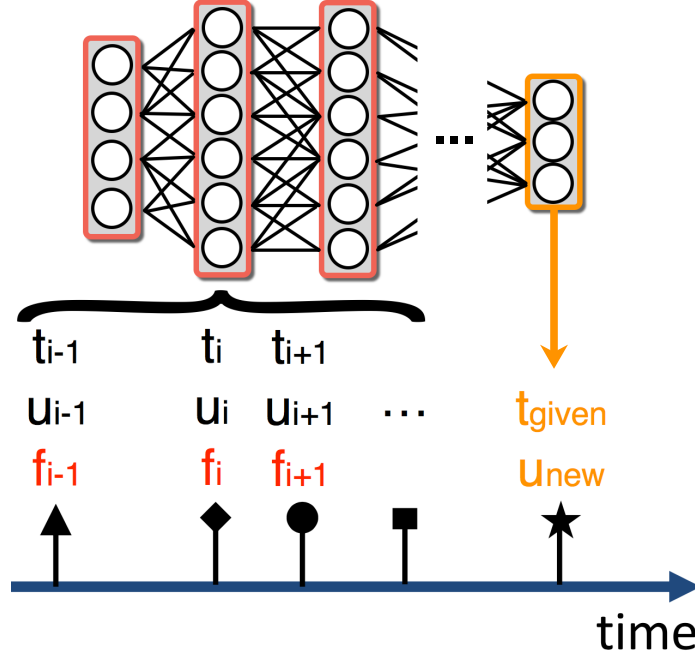


Figure 6.4: The discriminative learning of point processes implemented via a neural network.

and the intensity function is the composition of L functions, which can be achieved via L hidden layers of a neural network. The input of $F^1(\cdot)$ can be vectorized easily as the Section 6.3.1 does. The only assumption is that output of the L -th function $f^L(\cdot)$ should be non-negative, which ensures that our intensity function is physically-meaningful. In practice, we can apply exponential function $\exp(\cdot)$ or $\text{ReLU}(\cdot)_+$ to meet the requirement. The objective function is maximize the conditional probability of each event given time stamp, which corresponds to maximizing normalized intensity functions as:

$$\min_{\Theta} - \sum_i \log \left(\frac{\lambda_{c_i}(t_i)}{\sum_{c'} \lambda'_{c'}(t_i)} \right) + \mathcal{R}(\Theta), \quad (6.11)$$

where the first term corresponds to logistic regression if we apply exponential function as $f^L(\cdot)$. We can apply various regularizers $\mathcal{R}(\Theta)$ to our learning algorithm, and optimize the objective function via stochastic gradient descent method. Fig. 6.4 illustrates the scheme of our method.

6.4 Experiments

6.4.1 Baselines and Evaluations

Taking the following methods as baselines, we compare our method (named as DMCP) with them and demonstrate its superiority:

Markov chain (MC). Taking C CUs and D duration days as states, the simplest method is treating the event sequences as two independent Markov chains for the transition and the duration, respectively. Two one-order MCs are trained, whose transition matrices are calculated via counting the transitions among various states. In the prediction phase, given initial state (i.e., current CU and previous duration time), we use the transition matrices to predict next states (i.e., current duration time and next CU).

Vector auto-regressive model (VAR). Similar to the MC model, the VAR model used in this paper also captures the transitions among CUs and the durations in CUs as two independent transition processes, whose transition matrices are learned via the method in [5]. Different from the MC model, the transition matrix of the VAR model does not have probabilistic interpretation but is more flexible.

Continuous-time Markov Chain (CTMC). The CTMC [140], as a special type of semi-Markov model [17], also models the transition among CUs as a markov process. The transition process among CUs is modeled as a Markov chain in the continuous time domain, whose transition probability is time-varying. In the prediction phase, the destination CU is predicted according to previous CU and current transition matrix, and the duration in current CU is predicted via the interval between adjacent transitions.

Logistic regression (LR). Using the feature extracted from EHRs, we can treat the prediction of CU patient flow as two classification problems. Specifically, two multi-class classifiers are trained independently via multinomial logistic regression for destination CUs and duration days, respectively. In the training set, for each label c_i^n (or d_i^n), the feature is $[\mathbf{f}_0^{n\top}, \mathbf{f}_i^{n\top}]^\top$.

Hawkes processes (HP). Taking the transitions among CUs as event sequences, the parametric Hawkes process model in [20] is implemented, where the conditional intensity function is shown in Table 6.3. Different from our method, the Hawkes process is learned in a generative way — the likelihood of the whole event sequence is maximized via $\max_{\Theta} \prod_u \prod_i p(c_i^n, d_i^n | \mathcal{H}_t^n) (1 - P(T^n))$. In the prediction phase, given historical record \mathcal{H}_t^n , the predictions of next event (c, d) are obtained via $\max_{(c,d) \in \mathcal{C} \times \mathcal{D}} \int_{t+d-1}^{t+d} \lambda_c^n(s) ds$.

Modulated Poisson processes (MPP). The MPP replaces our mutually-correcting process with the modulated Poisson process [11] shown in Table 6.3. This method can be viewed as a point process-based interpretation of the generalized logit model of Markov chain.

Self-correcting process (SCP). Similar to the MPP, the SCP replaces our mutually-correcting process with the self-correcting process shown in Table 6.3.

The baselines above can be categorized into three classes: the MC, VAR, and CTMC methods are feature-independent, which merely rely on temporal information; the LR is history-independent, which merely relies on the EHR-based feature generated at current time while ignores historical record; the HP, MPP, SCP, including our DMCP are point process-based methods. The MPP, SCP, and our DMCP can be viewed as extensions of the LR method, which merge current features with historical ones via various point process models. To evaluate the performance of the pre-processing of imbalanced data, we consider our DMCP method with various pre-processing methods, including the **weighted data+DMCP (WDMCP)**, the **hierarchical data+DMCP (HDMCP)**, and the proposed **synthetic data+DMCP (SDMCP)**. The SCP with synthetic data (**SSCP**) is also tested to prove the universality of our data synthesis method.

The parameters in our DMHP method include the learning rate of gradient descent β , the bandwidth of Gaussian kernel σ in our mutually-correcting process model, and the weight of group-lasso γ . The learning rate β controls the step length of gradient descent. Too large β will lead our algorithm to be unstable while too small β will lead our algorithm

to converge too slowly. Following the work in [141], we set the learning rate β decays with rate $\mathcal{O}(k^{-1})$, where k is the number of iteration. Its initial value for our work is set as 10^{-4} . The parameter σ controls the importance of historical EHR-based features. When σ is large, the kernel $\exp(-\frac{(t-t')^2}{\sigma^2})$ decays slowly, which means the temporal influence of historical events will exist for a long time. In an extreme case that $\sigma \rightarrow \infty$, the kernel will tend to be 1, and our mutually-correcting process model will ignore the temporal difference among historical events and degrade to a self-correcting process. On the contrary, when σ is small, the kernel decays rapidly and the influence of historical events will be short. In the case that $\sigma \rightarrow 0$, our model will only consider the feature at current time and our learning algorithm will be similar to the LR method mentioned above. For achieving a trade-off, we set σ as the mean of duration days in our work. The parameter γ controls the importance of the group-lasso regularizer. In the case that the features of data indeed yield to the assumption of group sparsity, a suitable γ will regularize model well and improve the result of feature selection. Here, we set $\gamma = 1$ empirically.

Using the proposed data representation method, we can extract a large amount of feature-label pairs from event sequences, e.g., $(\mathbf{f}_{t_{i-1}}^n, c_i^n, d_i^n)$, where $\mathbf{f}_{t_{i-1}}^n$ is the feature of patient n containing her historical information before time t_{i-1} , c_i^n is her destination CU after t_{i-1} , and d_i^n is the duration time in c_i^n accordingly. Given these pairs, we evaluate all the methods above via 10-fold cross validation. Specifically, we use 90% of the data for training and the remaining 10% for testing randomly. The training data is further divided into 10 folds. For each method, its model is trained via 10 trials. In each trial, the 9-fold data is used to train the model while the remaining fold is for validation. The final model is the average of the 10 training results.

To evaluate various methods comprehensively, we apply the following measurements:

- **Prediction accuracy:** The prediction accuracy AC_c for each CU c and the overall

accuracy AC_C are calculated as

$$AC_c = \frac{\#\{\text{right prediction}\}}{\#\{\text{transitions to } c\}}, \quad AC_C = \sum_{c=1}^C \frac{\#\{\text{transitions to } c\}}{\#\{\text{total transitions}\}} AC_c.$$

The prediction accuracy AC_d for each duration category d and the overall accuracy AC_D are calculated in the same way.

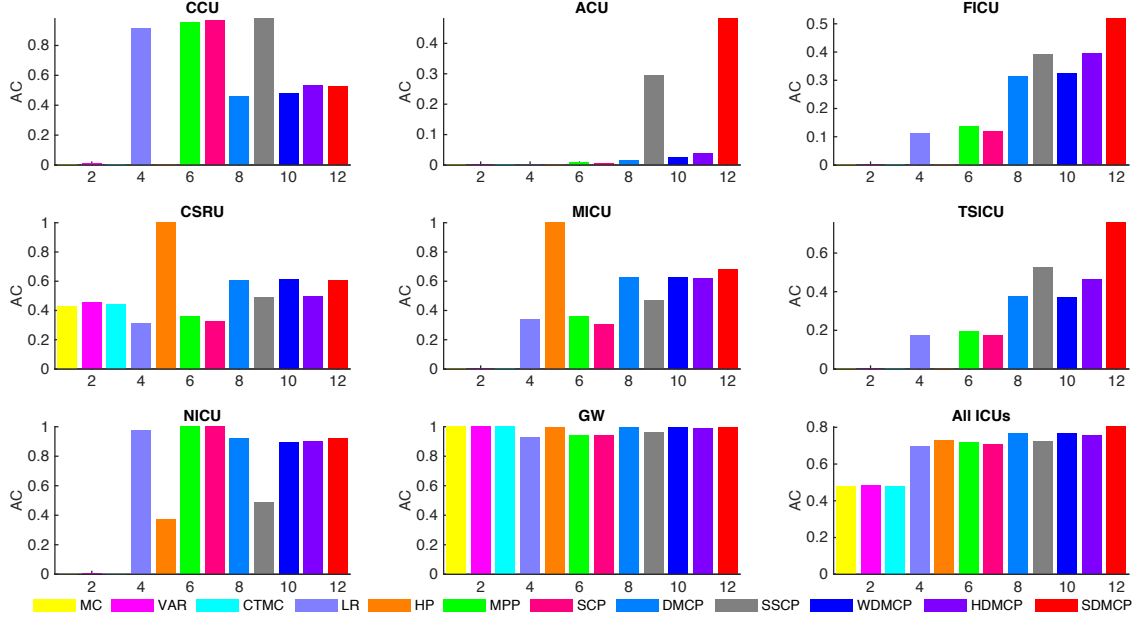
- **Relative simulation error:** Given historical patient data, we simulate the daily number of patients in each CU within the following week. The relative simulation error of patient flow Err_c for each CU c and the overall relative error Err_C are calculated as

$$Err_c = \frac{1}{7} \sum_{d=1}^7 \frac{|N_{c,d} - \hat{N}_{c,d}|}{N_{c,d}}, \quad Err_C = \frac{1}{7} \sum_{d=1}^7 \frac{|N_d - \hat{N}_d|}{N_d},$$

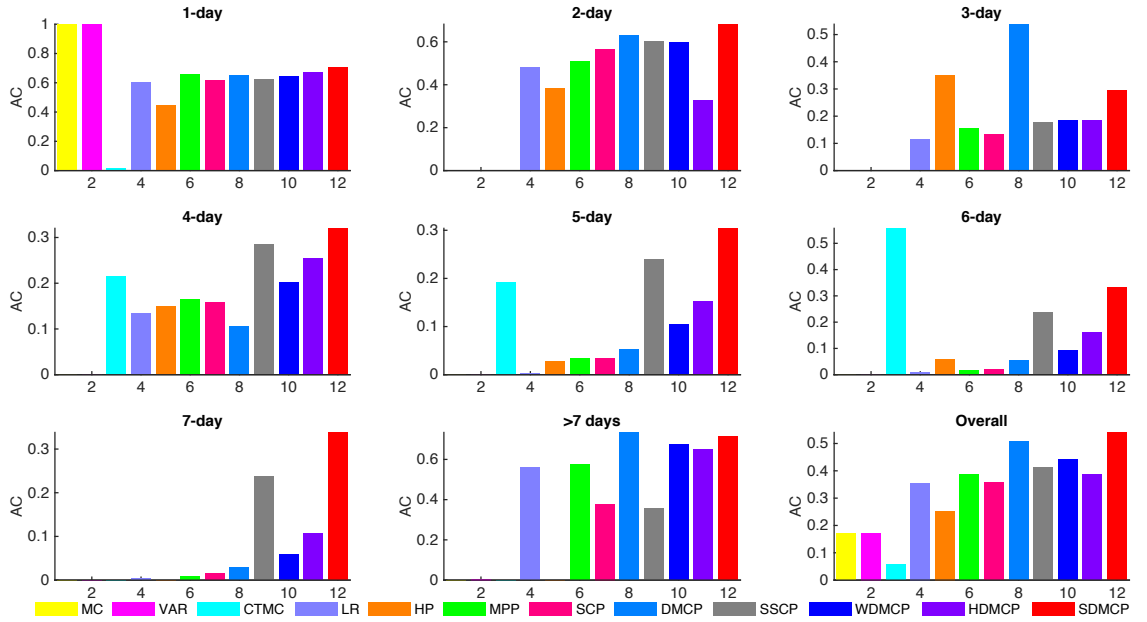
where $N_{c,d}$ (N_d) is the real number of patient in each CU (all CUs) in the d -th day, and $\hat{N}_{c,d}$ (\hat{N}_d) is the simulation result.

Table 6.4: Prediction accuracy for various methods on destination CUs

| Method | | AC_c 's | | | | | | | | AC_C |
|---------------------|-------|-----------|-------|-------|-------|-------|-------|-------|-------|---------|
| | | CCU | ACU | FICU | CSRU | MICU | TSICU | NICU | GW | All CUs |
| Without pre-process | MC | 0 | 0 | 0 | 0.430 | 0 | 0 | 0 | 1.000 | 0.478 |
| | VAR | 0.010 | 0 | 0.002 | 0.455 | 0 | 0 | 0.005 | 1.000 | 0.483 |
| | CTMC | 0 | 0 | 0 | 0.438 | 0 | 0 | 0 | 1.000 | 0.479 |
| | LR | 0.912 | 0.002 | 0.113 | 0.311 | 0.338 | 0.172 | 0.977 | 0.932 | 0.696 |
| | HP | 0 | 0 | 0 | 0.999 | 0.997 | 0 | 0.372 | 0.996 | 0.731 |
| | MPP | 0.954 | 0.008 | 0.138 | 0.359 | 0.356 | 0.194 | 0.995 | 0.942 | 0.719 |
| | SCP | 0.965 | 0.006 | 0.117 | 0.323 | 0.301 | 0.171 | 0.997 | 0.940 | 0.705 |
| | DMCP | 0.461 | 0.014 | 0.313 | 0.605 | 0.628 | 0.376 | 0.924 | 0.995 | 0.766 |
| With pre-process | SSCP | 0.979 | 0.295 | 0.393 | 0.486 | 0.470 | 0.525 | 0.486 | 0.963 | 0.724 |
| | WDMCP | 0.479 | 0.025 | 0.325 | 0.612 | 0.629 | 0.371 | 0.894 | 0.995 | 0.766 |
| | HDMCP | 0.534 | 0.040 | 0.396 | 0.496 | 0.619 | 0.466 | 0.901 | 0.991 | 0.758 |
| | SDMCP | 0.529 | 0.482 | 0.520 | 0.606 | 0.684 | 0.758 | 0.920 | 0.995 | 0.805 |



(a) The prediction accuracy of destination CUs obtained via various methods.



(b) The prediction accuracy of duration days obtained via various methods.

Figure 6.5: (a) The prediction accuracy of each CUs and the overall accuracy are given. In each subfigure, the color bars correspond to various learning methods. (b) The prediction accuracy of each duration day and the overall accuracy are given. In each subfigure, the color bars correspond to various learning methods.

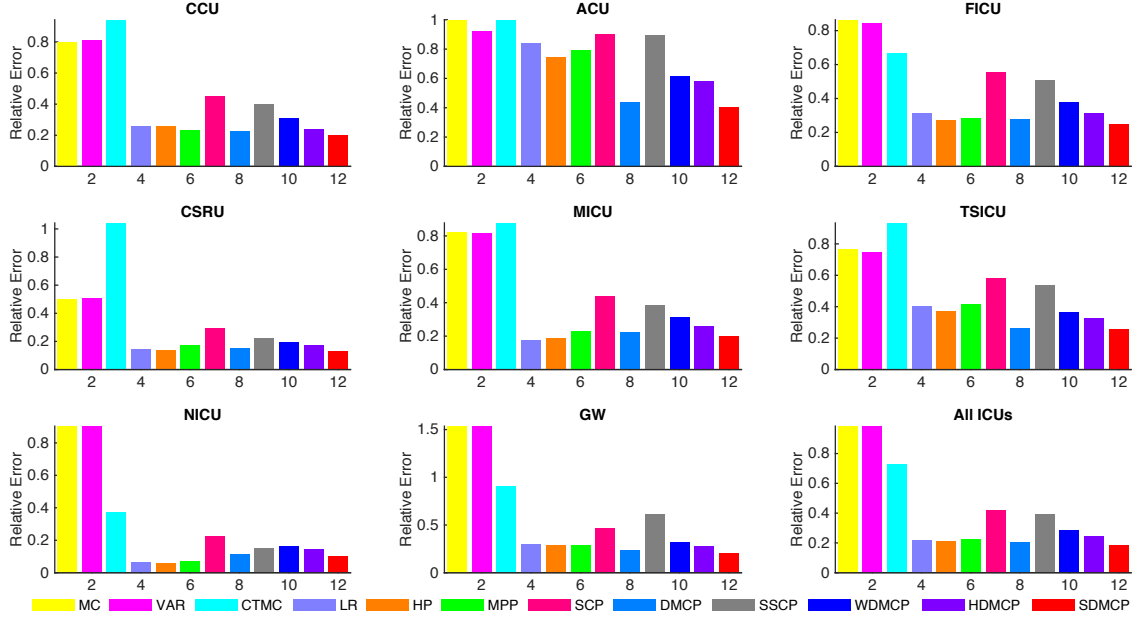


Figure 6.6: The relative simulation error of each CU and the overall simulation error are given. In each subfigure, the color bars correspond to various learning methods.

Table 6.5: Prediction accuracy for various methods on duration days

| Method | | AC _c 's | | | | | | | | AC _C |
|---------------------|-------|--------------------|-------|-------|-------|-------|-------|-------|--------|-----------------|
| | | 1-day | 2-day | 3-day | 4-day | 5-day | 6-day | 7-day | >7-day | Overall |
| Without pre-process | MC | 1.000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.171 |
| | VAR | 1.000 | 0.002 | 0 | 0 | 0 | 0 | 0 | 0.005 | 0.173 |
| | CTMC | 0.017 | 0 | 0 | 0.215 | 0.193 | 0.557 | 0 | 0 | 0.058 |
| | LR | 0.603 | 0.481 | 0.115 | 0.134 | 0.004 | 0.008 | 0.003 | 0.560 | 0.355 |
| | HP | 0.445 | 0.385 | 0.350 | 0.150 | 0.029 | 0.058 | 0 | 0 | 0.253 |
| | MPP | 0.656 | 0.509 | 0.158 | 0.165 | 0.035 | 0.019 | 0.009 | 0.574 | 0.387 |
| | SCP | 0.618 | 0.568 | 0.135 | 0.158 | 0.035 | 0.022 | 0.016 | 0.378 | 0.357 |
| | DMCP | 0.652 | 0.633 | 0.538 | 0.105 | 0.052 | 0.054 | 0.029 | 0.734 | 0.508 |
| With pre-process | SSCP | 0.628 | 0.605 | 0.178 | 0.285 | 0.240 | 0.238 | 0.237 | 0.357 | 0.412 |
| | WDMCP | 0.646 | 0.599 | 0.185 | 0.204 | 0.106 | 0.097 | 0.059 | 0.676 | 0.443 |
| | HDMCP | 0.677 | 0.329 | 0.187 | 0.256 | 0.154 | 0.162 | 0.109 | 0.652 | 0.389 |
| | SDMCP | 0.705 | 0.683 | 0.297 | 0.321 | 0.305 | 0.335 | 0.340 | 0.715 | 0.542 |

6.4.2 Comparison Results

We compare our DMCP method with its competitors on predicting destination CUs and duration days in current CUs, and simulating the patient flow in the future. The prediction results are shown in Fig. 6.5, and the relative simulation errors are shown in Fig. 6.6.

Table 6.6: Overall prediction accuracy for various methods on relative simulation errors

| Method | | Err _c 's | | | | | | | | Err _C |
|------------------------|-------|---------------------|-------|-------|-------|-------|-------|-------|-------|------------------|
| | | CCU | ACU | FICU | CSRU | MICU | TSICU | NICU | GW | All CUs |
| Without pre-process | MC | 0.799 | 1.002 | 0.864 | 0.504 | 0.821 | 0.767 | 0.903 | 1.536 | 0.984 |
| | VAR | 0.803 | 0.903 | 0.861 | 0.506 | 0.819 | 0.760 | 0.903 | 1.535 | 0.982 |
| | CTMC | 0.942 | 1.003 | 0.664 | 1.042 | 0.877 | 0.933 | 0.373 | 0.908 | 0.730 |
| | LR | 0.256 | 0.838 | 0.311 | 0.148 | 0.196 | 0.407 | 0.064 | 0.294 | 0.215 |
| | HP | 0.247 | 0.855 | 0.329 | 0.156 | 0.191 | 0.393 | 0.059 | 0.296 | 0.218 |
| | MPP | 0.230 | 0.790 | 0.287 | 0.172 | 0.231 | 0.420 | 0.068 | 0.293 | 0.224 |
| | SCP | 0.453 | 0.903 | 0.553 | 0.295 | 0.442 | 0.580 | 0.227 | 0.464 | 0.419 |
| | DMCP | 0.230 | 0.468 | 0.281 | 0.142 | 0.223 | 0.273 | 0.114 | 0.230 | 0.204 |
| With pre-process | SSCP | 0.398 | 0.894 | 0.508 | 0.224 | 0.386 | 0.538 | 0.150 | 0.610 | 0.395 |
| | WDMCP | 0.316 | 0.611 | 0.357 | 0.206 | 0.308 | 0.386 | 0.149 | 0.323 | 0.281 |
| | HDMCP | 0.290 | 0.433 | 0.288 | 0.168 | 0.259 | 0.398 | 0.141 | 0.277 | 0.243 |
| | SDMCP | 0.201 | 0.406 | 0.245 | 0.131 | 0.197 | 0.259 | 0.100 | 0.208 | 0.181 |

The numerical results of overall prediction accuracy and simulation error are shown in Tables 6.4, 6.5, and 6.6. Experimental results of these three tasks show that our DMCP method obtains superior results in most situations and outperforms other methods. Furthermore, adding proposed data synthesis method as the pre-processing of training data, our SDMCP method further improves the prediction results. Specifically, we can find that:

1) According to Fig. 6.5, Fig. 6.6, and Tables 6.4, 6.5, we can find that our DMCP methods obtain the highest overall prediction accuracy and the lowest simulation error. Compared with the second best methods, i.e., the HP for predicting destination CUs and the MPP for predicting duration days, our DMCP achieves improvements over 4% and 11% respectively. These encouraging results demonstrate that our mutually-correcting process model is suitable for describing patient flow.

2) The feature-independent methods (MC, VAR and CTMC) perform poorly in all three tasks. Because of the imbalance of data, there are insufficient transition processes involving those rarely-used CUs. As a result, the transition probabilities learned via MC and CTMC and the transition coefficients learned via VAR are unreliable. For example, in Fig. 6.5(a), we can find that these methods only obtain high accuracy when predicting general ward because it is contained via most patients' transition processes. However, the prediction

accuracy of other CUs is almost zero in most situations. Similar phenomenon can also be observed in the prediction results of duration days — only the 1-day situation is predicted with high accuracy while the rest situations cannot be predicted.

3) Compared with feature-independent methods, the LR method improves the testing results greatly, which demonstrates the importance of EHR-based features for predicting patient flow. Specifically, in Fig. 6.5 we can find that LR outperforms MC, VAR, and CTMC in most situations, whose overall accuracy is improved over 20% in both prediction tasks.

4) The point process-based methods (HP, MPP, SCP, and our DMCP) further improve the prediction accuracy of both two learning tasks because of considering the temporal influences of historical features on current predictions. Specifically, the HP method trains a Hawkes process model in a generative way, and the joint probability $p(c, d, t | \mathcal{H}_t^n)$ is estimated. However, as aforementioned, such a generative learning method is sensitive to the insufficiency and imbalance of data. As a result, the predictive model does not work when it comes to predict the classes having few samples (i.e., ACU, FICU, and TSICU in Fig. 6.5(a) and the duration with 7-day in Fig. 6.5(b)). On the contrary, the discriminative learning methods (MPP, SCP and our DMCP) are more robust, which improves the prediction accuracy for the classes having few samples.

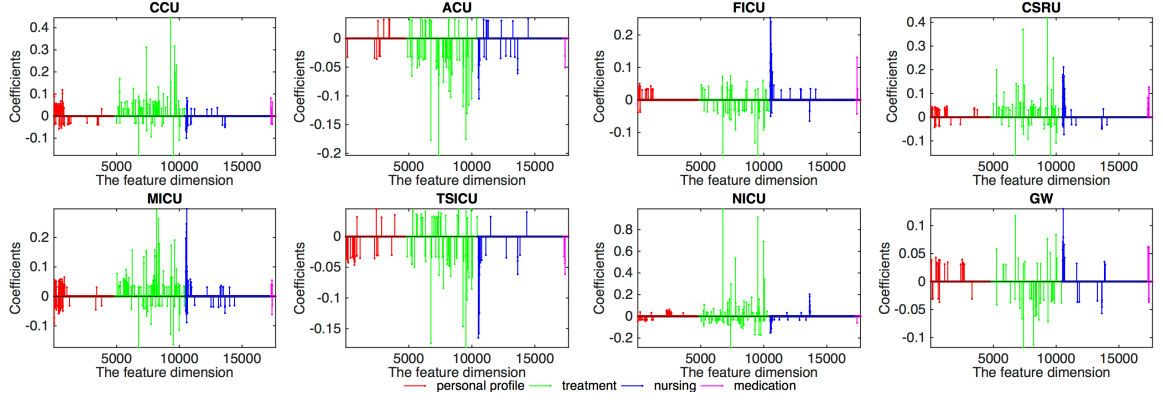
5) Adding suitable pre-processing in the training phase indeed enhances the robustness of our DMCP method to imbalanced data. In Tables 6.4, 6.5, and 6.6, we can find that because of the weaknesses analyzed in Section 6.3.3, WDMCP and HDMCP are slightly inferior to original DMCP method. Fig. 6.5 illustrates the reason obviously: while the prediction accuracy for those minor classes (i.e., the ACU, FICU in Fig. 6.5(a) and the 4-day in Fig. 6.5(b)) is improved, the performance on major classes degrades more (i.e., the CSRU, NICU in Fig. 6.5(a) and the 2-and 3-day in Fig. 6.5(b)). The proposed SDCMP, on the contrary, improves the result of minor classes and avoids the degradation of the result of major classes jointly, which obtains even better results than original DMCP — both the

AC_C and AC_D increase over 3% and the Err_C is reduced to 0.183.

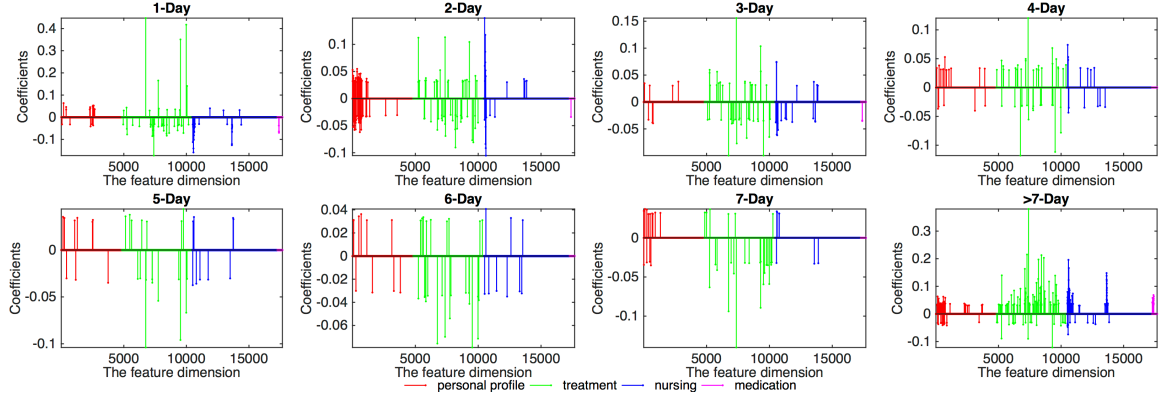
6.4.3 Feature Selection Result

As aforementioned, our method achieves feature selection via the group-lasso regularizer. Treating each dimension of feature as a group, we measure the importance of each group via the amplitude of the coefficient associated with the group, denoted as $|\Theta_m|$. The large amplitude means that the change of feature corresponding to the coefficient has a huge influence on the prediction result. In particular, when the coefficient is positive, it means that the corresponding feature will increase the conditional intensity function. Such a feature (profile, treatment, nursing operation, or medication) increases the probability that transitioning patients to certain CUs and staying certain days. On the contrary, when the coefficient is negative, the corresponding feature decreases the probability of certain transition events. When the coefficient is zero, it means that the corresponding feature does not change the conditional intensity function, and therefore, does not have any contribution to predict patient flow.

Figs. 6.7(a) and 6.7(b) visualize the coefficients in different feature domains w.r.t. various learning tasks. We can find that the time-varying features related to treatments are with dense coefficients while the time-invariant features (personal profile) and the time-varying features related to nursing programs and medications are with very sparse coefficients. Additionally, many features have negative coefficients. It means that these features suppress the transitions among CUs and lengthen the duration in current CU. We think these phenomena are reasonable based on the following reasons. 1) The treatments are the most influential factors for the patient flow, whose progresses and feedbacks impact on the transitions between CUs and the durations in them greatly. Therefore, it is natural that the features in this domain are with large coefficients. 2) The features in the personal profile domain are likely to be correlated with each other. For example, a certain disease's diagnose is correlated with patient's age and gender. Therefore, only a part of features in this



(a) The distribution of coefficients for predicting destination CUs.



(b) The distribution of coefficients for predicting duration days.

Figure 6.7: Feature selection result.

domain are selected. 3) Similarly, nursing programs and medications are highly correlated with treatments. When most of the features related to treatments are selected, only a part of them are useful. 4) Some diseases and corresponding treatments require patients to stay at certain CUs for a long time. When the treatments, nursing operations, or medications are applied, the patients are unlikely to transit to other CUs in few days.

6.5 Summary

In this chapter, we propose a flexible mutually-correcting process model to capture the properties of patient flow and apply a discriminative algorithm to learn the model. The connection between our method and deep learning techniques is discussed. Focusing on predicting patient flow, we apply our discriminative learning method and demonstrate its

usefulness. In this task, our mutually-correcting process model improves the flexibility of existing parametric point process models, which reflects the properties of patient flow. The proposed discriminative learning algorithm combines multinomial logistic regression with the group-lasso regularizer, and achieves feature selection during learning model. We also consider the data imbalance problem in the real-world dataset and propose a novel pre-processing method for training samples, which greatly improves the learning result. Compared with the state-of-art methods, our method obtains superior prediction results on real-world data set, which has potential to predict overcrowdedness or conflicted usage of CUs in practical situations. Our method is applicable to modeling a patient's need for various "care teams" within the CU (critical care nurses, a pharmacist, a nutritionist, respiratory therapists, consultants, social workers and case managers, clergy, etc), which will further improve care management and coordination for patients with multiple chronic conditions.

CHAPTER 7

POINT PROCESS-BASED TRAILER GENERATION

7.1 Introduction

7.1.1 Motivations

In the previous chapters, the applications of our point process models are mainly about data analysis and data mining, e.g., causality analysis of IPTV data, clustering users according to their behaviors, constructing disease networks, and predicting patient flow. However, the application of point processes should not be limited to data mining — actually, we can extend their applications to more fields. In this chapter, we propose an innovative application of our point process model in the field of computer vision, modeling the attractiveness of video clips and generating movie trailer automatically.

With the proliferation of online video sites such as Youtube, promoting online videos through advertisements is becoming more and more popular and important. Nowadays, most advertisements consist of only key frames or shots accompanied by textual descriptions. Although these advertisements can be easily produced using existing video summarization techniques, they are oftentimes not attractive enough. Only a small portion of videos, e.g., Hollywood movies, are promoted by highly attractive trailers consisting of well-designed montages and mesmerizing music. Nevertheless, the quality of trailers generated by video summarization techniques are far from satisfactory, and it is reasonable because the goal of trailer generation is to maximize the video attractiveness, or equally, to minimize the loss of attractiveness, whereas video summarization aims at selecting key frames or shots to capture storylines or important scenes. To produce highly attractive trailers, human expertise and creativity are always needed, making trailer generation procedures costly. In this chapter, we study how to produce trailers automatically and efficiently, and

our approach may be applied to potentially millions of online videos and hence lower the cost substantially.

Trailer generation is challenging because we not only need to select key shots but also re-organize them in such a coherent way that the whole trailer is most attractive. Nevertheless, attractiveness is a relatively ambiguous and subjective notion, and it may be conveyed through several factors such as the shots, the order of shots, and the background music. In this paper, we propose an automatic trailer generation approach which consists of three key components: *1) a practical surrogate measure of trailer attractiveness; 2) an efficient algorithm to select and re-organize shots to maximize the attractiveness; 3) an effective method to synchronize shots and music for improved viewer experience.* Specifically, we learn an attractiveness model for movie trailers by leveraging the self-correcting point process methodology in [24, 25]. Then, we position the trailer montages by exploiting the saliency information of the theme music. Finally, based upon the montage information and the attractiveness model, we construct a shot graph and generate a trailer by finding the shortest path that is equivalent to minimize the loss of attractiveness.

We summarize our contributions as follows: 1) We propose an effective surrogate measure of video attractiveness, namely, fixation variance. With this measure, we study the dynamics of video attractiveness and the properties of movie trailer. 2) Although point processes have been widely used for modeling temporal event sequences, such as earthquakes [25] and social behaviors [34], to the best of our knowledge, our methodology is the first to model video attractiveness using self-correcting point processes. 3) We investigate the influence of music on trailer generation, and propose a graph-based music-guided trailer generation method. Compared to the state-of-the-art methods, our method achieves significantly better results while using much shorter time.

7.1.2 Background

Video summarization techniques have attracted much research interest and many methods have been proposed in the past. The early work in [142, 143] extracts features of frames and clusters frames accordingly, but their performance is limited because only visual features are considered. Besides visual frames, other information has been taken into consideration in video summarization, including the attentions [144, 145] and the interactions [146] of viewers, the textual information [147], and the aural information [148]. Recently, a joint aural, visual, and textual attention model is proposed for movie summarization in [149]. Moreover, semantic information of videos has been exploited, including the saliency maps of images [150], special events [151, 152], key people and objects [153, 154], and storylines [155]. The external information sources such as web images have also been demonstrated to be useful [154, 155].

Focusing on trailer generation problem, a user attention model is proposed for trailer generation in [156]. A trailer generator combining Plutchik’s emotion model [157, 158] with Bayesian surprise model [159] is proposed in [160], which achieves the state-of-the-art performance. However, the system does not incorporate the relationships between a trailer and its music, and the causality between the surprise degree and video attractiveness is questionable. The definition and the measurement of visual attractiveness are largely overlooked in these methods, which turns out to be rather critical for trailer generation.

7.2 Properties of Movie Trailer

Suppose that we have a set of K training trailers $\{\mathcal{T}_k\}_{k=1}^K$, and in total N shots $\{\mathcal{C}_i\}_{i=1}^N$. Each shot comes from one trailer and consists of a set of frames. Let $\mathcal{C}_i \in \mathcal{T}_k$ indicate that shot \mathcal{C}_i comes from the trailer \mathcal{T}_k , and $\mathcal{C}_i = \{f_j^{(i)}\}_{j=1}^{n_i}$ indicate that there are n_i frames in \mathcal{C}_i and $f_j^{(i)}$ is the j th frame of \mathcal{C}_i . Similarly, we use $\mathcal{T} \subset \mathcal{V}$ to indicate that \mathcal{T} is the trailer of the video \mathcal{V} . We also represent a video \mathcal{V} or a trailer \mathcal{T} as a sequence of shots, denoted as

$\{\mathcal{C}_i\}_{i=1}^N$, in the sequel. We use the index of a frame as the time stamp of the shot (trailer and movie) for convenience. The beginning and the ending of a video or a trailer are denoted as $L_0 = 0$ and $L_N = \sum_{i=1}^N n_i$. The position of montage between \mathcal{C}_i and \mathcal{C}_{i+1} is denoted as $L_i = L_{i-1} + n_i = \sum_{j=1}^i n_j$. The *trailer generation problem* is: given a video \mathcal{V} and a piece of music \mathbf{m} , we would like to generate a trailer $\mathcal{T} \subset \mathcal{V}$ that is the most attractive.

7.2.1 Measure and Dynamics of Attractiveness

We might observe such a common phenomenon: when attractive scenes such as handsome characters and hot actions appear, viewers will look at the same area on the screen; on the other hand, when boring scenes such as the cast of characters and tedious dialogues appear, viewers will no longer focus on the same screen area. In other words, the attractiveness of a video is highly correlated with the attention of viewers when they watch the video. Therefore, we propose a surrogate measure of attractiveness based on viewers' eye-movement, whose efficacy is validated by the following experiments. Specifically, we invite 14 (6 female and 8 male) volunteers to watch 8 movie trailers, which contain 1,083 shots¹. We further record the motions of their gazes and calculate the mapped fixation points in each frame using Tobii T60 eye tracker. Denote the locations of gaze on the screen, namely, the fixation points, in the j -th frame of \mathcal{C}_i as $[\mathbf{x}_j^{(i)}, \mathbf{y}_j^{(i)}]$, where $\mathbf{x}_j^{(i)} \in \mathbb{R}^{14}$ (resp. $\mathbf{y}_j^{(i)} \in \mathbb{R}^{14}$) is the vector of the horizontal (resp. vertical) coordinates of the fixation points of the 14 volunteers. For the j -th frame, we define the *fixation variance* as the determinant of the covariance matrix of the fixation points:

$$\sigma_j^{(i)} = \det(\text{cov}([\mathbf{x}_j^{(i)}, \mathbf{y}_j^{(i)}])), \quad (7.1)$$

We average $\{\sigma_j^{(i)}\}_{j=1}^{n_i}$ for the frames belonging to the same shot. The averaged fixation variance reflects the spread of attention when watching the shot. Following the above reasoning and definition, we expect that the boring shots (e.g., background) should have large

¹In this paper, we segment video into shots using a commercial software "CyberLink PowerDirector".

fixation variance whereas the attractive shots (e.g., hot action scenes, characters) should have small fixation variance. To verify this, we label these two types of shots manually and calculate the statistics of their fixation variance. The results are summarized in Table 7.1.

Table 7.1: The statistics of normalized fixation variance ($\times 10^8$)

| | mean(σ) | median(σ) | variance(σ) |
|------------------|------------------|--------------------|----------------------|
| Boring shots | 1.19 | 0.45 | 0.03 |
| Attractive shots | 0.60 | 0.22 | 0.01 |

It is easy to observe that both the mean and the median of the fixation variance of boring shots are about twice larger than those of the attractive shots. The variance is very small, meaning that our proposed fixation variance is stable in both boring and attractive shot groups. These results show that fixation variance is negatively correlated with the video attractiveness — it measures the loss of attractiveness accurately and robustly. Fig. 7.1 further shows typical examples.

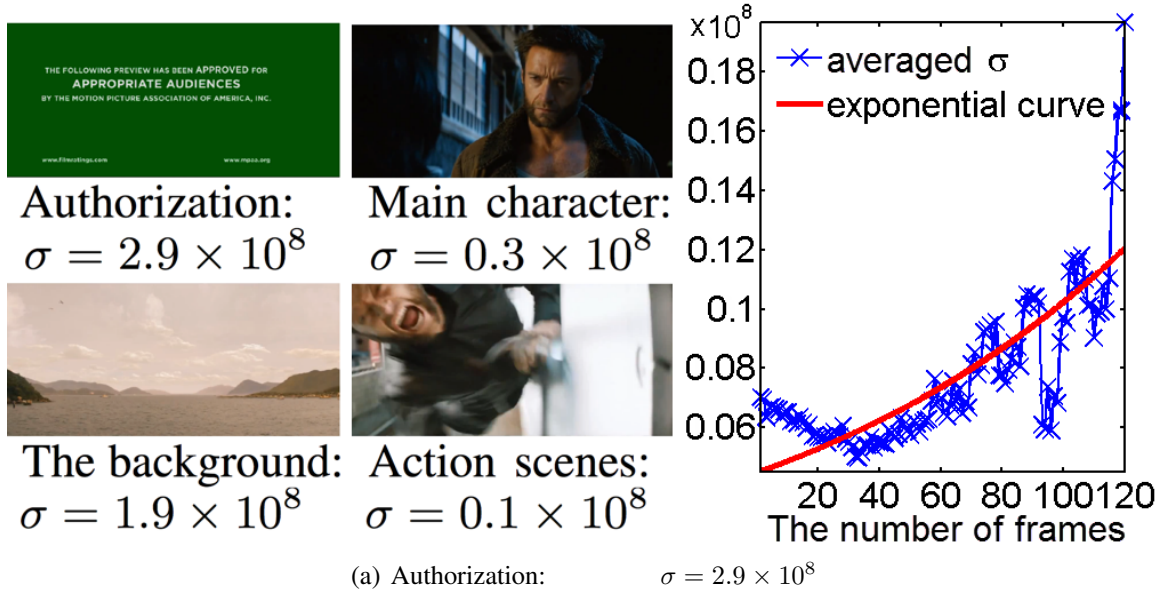


Figure 7.1: The left subfigure shows four shots in trailer “*The Wolverine 2013*” and their fixation variances. The right subfigure shows dynamics of fixation variance calculated from training trailers.

The dynamics of fixation variance. Given N shots $\{\mathcal{C}_i\}_{i=1}^N$, the averaged fixation variance in the j th frame is calculated as $\bar{\sigma}_j = \frac{1}{N_j} \sum_{i=1}^{N_j} \sigma_j^{(i)}$, where N_j is the number

of shots having at least j frames. In Fig. 7.1, we find that the change of $\bar{\sigma}_j$ over time can be approximated by an increasing exponential curve. In other words, within a shot, its attractiveness decreases over time. The inter-shot dynamics can be modeled as the stitching of the fitted exponential curves for adjacent shots. It means that although the attractiveness within one shot decreases over time, the montage between shots increases attractiveness.

7.2.2 The Dynamics of Music

Similar to the dynamics of attractiveness, we empirically find that the dynamics of music are also highly correlated with the montages between shots. To see this, we first detect the saliency points of the music associated with a trailer as follows. 1) Using the saliency detection algorithm in [161], we extract the saliency curve of a piece of music as

$$\hat{\mathbf{m}} = G((\text{idct}(\text{sign}(\text{dct}(\mathbf{m}))))^2), \quad (7.2)$$

where $\text{dct}(\cdot)$ and $\text{idct}(\cdot)$ are a DCT transformation pair, $\text{sign}(\cdot)$ is the sign function that returns 1, -1, and 0 for positive, negative, and zero inputs, respectively. $G(\cdot)$ is a Gaussian filter. 2) After re-sampling $\hat{\mathbf{m}}$ with the number of frames, we detect the peaks in $\hat{\mathbf{m}}$. Regarding these peaks as the saliency points of the music, we investigate their correlations with the montages in the trailer as follows. For each peak, we label the peak as a correct indicator of the location of the montage when a montage appears within ± 6 frames (about 0.25 second). On the 8 sample trailers, we find that the time stamps of the saliency points are highly correlated with those of the montages (with accuracy 84.88%). Fig. 7.2 presents an example: in high-quality trailers, the montages of shots are synchronized with the rhythm of the background music.

According to the analytic experiments above, we summarize three properties of movie trailer as follows: **Property 1.** *The loss of attractiveness can be approximated by a surrogate measure, namely, fixation variance.* **Property 2.** *Within each shot, the loss of attrac-*

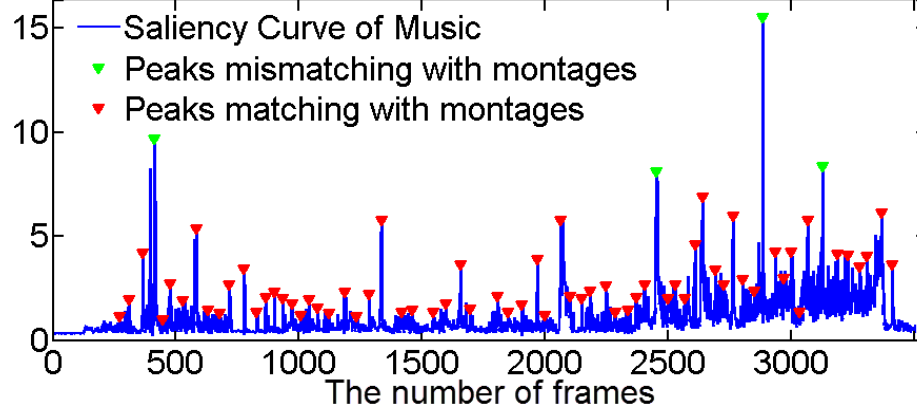


Figure 7.2: The Saliency Points of Music v.s. Montage Positions (Trailer of “*The Bling Ring*”).

tiveness increases exponentially and this tendency is corrected when a new shot appears.

Property 3. *The self-correction of attractiveness, the rhythm of the music in the video, and the montages between shots are highly correlated.*

7.3 Point Process-based Attractiveness Model

7.3.1 Motivation

Our modeling assumption is that the fixation variance is highly correlated with the number of viewers losing their attention on the screen, which directly connects the notion of the attractiveness of a video with a specific point process model we will discuss below. To this end, suppose that there are V viewers watching a movie. We define a sequence $E_v(t)$ of the event “whether the viewer loses her attention or not at time t ” for each viewer v :

$$E_v(t) = \begin{cases} 1, & \text{viewer } v \text{ loses her attention at time } t, \\ 0, & \text{otherwise.} \end{cases} \quad (7.3)$$

Although we do not observe the event sequence directly, we assume that the fixation variance is proportional to the number of viewers losing their attention. Therefore, given the fixation variance of training trailers, we can approximate the aggregated observations of

viewers' events $\sum_v E_v(t)$, and model viewers' events as a temporal point process. Specifically, we propose an attractiveness model based on a specific point process, i.e., self-correcting point process.

7.3.2 Self-Correcting Point Process

A self-correcting point process is a point process with the following intensity function:

$$\lambda(t) = \frac{\mathbb{E}(dN(t)|\mathcal{H}_t)}{dt} = \exp\left(\alpha t - \sum_{i:t_i < t} \beta\right), \quad (7.4)$$

where $N(t)$ is the number of events occurred in time range $(-\infty, t]$, \mathcal{H}_t denotes the historical events happened before time t , and $\mathbb{E}(dN(t)|\mathcal{H}_t)$ is the expectation of the number of events happened in the interval $(t, t + dt]$ given historical observations \mathcal{H}_t . The intensity function in Eq. (7.4) represents the expected instantaneous rate of future events at time t .

The intensity function of the self-correcting point process increases exponentially with rate α and this tendency can be corrected by the historical observations via rate β . Note that the intensity function exactly matches the dynamics of attractiveness described in Property 2. Therefore, given a video $\mathcal{V} = \{\mathcal{C}_i\}_{i=1}^N$, for each shot \mathcal{C}_i , we define the *local* intensity function in its time interval $(0, n_i]$ as

$$\lambda_{\mathcal{C}_i}(t) = \exp\left(\alpha_i H^i t - \beta_i D_t^i\right), \quad (7.5)$$

where $t \in (0, n_i]$ and

$$H^i = \begin{cases} H(\hat{f}_1^{(1)}), & i = 1, \\ D(\hat{f}_{N_{i-1}}^{(i-1)} || \hat{f}_1^{(i)}), & i > 1, \end{cases} \quad (7.6)$$

$$D_t^i = \sum_{j=2}^{\lfloor t \rfloor} D(\hat{f}_{j-1}^{(i)} || \hat{f}_j^{(i)}), \quad (7.7)$$

where $\hat{f} = G((\text{idct2D}(\text{sign}(\text{dct2D}(f))))^2)/C$ is the normalized saliency map of frame f [161]. C is a l_1 normalizer that guarantees \hat{f} to be a distribution. For the first shot \mathcal{C}_1 of \mathcal{V} ($i = 1$), $H^i = H(\hat{f}_1^{(1)})$ represents the entropy of the first frame in \mathcal{C}_1 , which is the *initial stimulus* given by \mathcal{C}_1 . For the following shots ($i > 1$), $H^i = D(\hat{f}_{N_{i-1}}^{(i-1)} || \hat{f}_1^{(i)})$ represents the KL-divergence between the last frame of \mathcal{C}_{i-1} and the first frame of \mathcal{C}_i , which is the *initial stimulus* given by \mathcal{C}_i . Similarly, $D(\hat{f}_{j-1}^{(i)} || \hat{f}_j^{(i)})$ represents the KL-divergence between the adjacent frames in \mathcal{C}_i , which is the *supplementary stimulus*. $D_t^i = \sum_{j=2}^{\lfloor t \rfloor} D(\hat{f}_{j-1}^{(i)} || \hat{f}_j^{(i)})$ is the accumulative influence caused by the supplementary stimulus in the time interval $(0, t]$, where $\lfloor t \rfloor$ is the largest integer smaller than t .

The intensity function in Eq. (7.5) imitates the loss of attractiveness — it increases exponentially with the initial stimulus till the supplementary stimulus corrects this tendency. For each shot \mathcal{C}_i , its intensity function has two non-negative parameters (α_i, β_i) . To summarize, we define our attractiveness model as a self-correcting point process with a *global* intensity function for time interval $(0, \sum_{i=1}^N n_i]$, which stitches N local intensity functions as $\lambda_{\mathcal{V}}(t) = \sum_{i=1}^N \lambda_{\mathcal{C}_i}(t - L_{i-1})$, where $L_0 = 0$, $L_i = L_{i-1} + n_i = \sum_{j=1}^i n_j$.

7.3.3 Model Learning

Given K training trailers $\{\mathcal{T}_k\}_{k=1}^K$ consisting of N shots, our goal is to learn parameters of the N *local* intensity functions. Similar to [34], we achieve this goal by pursuing an maximum likelihood estimation (MLE) of the parameters, and the likelihood function can be written as:

$$\mathcal{L} = \prod_{i=1}^N \mathcal{L}_i = \prod_{i=1}^N \left(\left(\prod_{t=1}^{n_i} (\lambda_{\mathcal{C}_i}(t))^{\sigma_{\mathcal{C}_i}(t)} \right) \exp \left(-\sigma_m \int_0^{n_i} \lambda_{\mathcal{C}_i}(s) ds \right) \right),$$

where L_i denotes the local likelihood for \mathcal{C}_i , $\sigma_{\mathcal{C}_i}(t)$ is the fixation variance of the t th frame of shot \mathcal{C}_i , σ_m is the maximum of fixation variance.

Besides capturing the *global* event dynamics by maximizing Eq. (7.8), we also require the proposed model to fit *local* event information in each time interval. Therefore, we further propose to minimize a novel data fidelity loss function to correlate the local intensity and fixation variance in each frame: $\sum_{i=1}^N \sum_{t=1}^{n_i} |\log(\gamma \lambda_{\mathcal{C}_i}(t) / \sigma_{\mathcal{C}_i}(t))|^2$, where γ is shared by all frames. This term encourages the local intensity (scaled by γ) to be equal to the fixation variance in each frame. To sum up, we learn our model by solving the following problem:

$$\begin{aligned} \min_{\alpha, \beta, \gamma} & -\log(L) + \mu \sum_{i=1}^N \sum_{t=1}^{n_i} \left| \log \left(\frac{\gamma \lambda_{\mathcal{C}_i}(t)}{\sigma_{\mathcal{C}_i}(t)} \right) \right|^2, \\ \text{s.t. } & \alpha \geq \mathbf{0}, \beta \geq \mathbf{0}, \gamma > 0, \end{aligned}$$

where $\alpha = [\alpha_1, \dots, \alpha_N]$ and $\beta = [\beta_1, \dots, \beta_N]$ represent the parameters of N local intensity functions.

We develop an alternating algorithm to solve Eq. (7.8). To be specific, given the initial values of (α, β, γ) , we first solve the following subproblem for each shot with γ fixed:

$$\begin{aligned} \min_{\alpha_i, \beta_i} & -\log(L_i) + \mu \sum_{t=1}^{n_i} \left| \log \left(\frac{\gamma \lambda_{\mathcal{C}_i}(t)}{\sigma_{\mathcal{C}_i}(t)} \right) \right|^2 \\ \text{s.t. } & \alpha_i \geq 0, \beta_i \geq 0. \end{aligned}$$

The objective of Eq. (7.8) can be written as follows:

$$\begin{aligned} I_i = & \sum_{t=1}^{n_i} \left(\sigma_{\mathcal{C}_i}(t) (\beta_i D_t^i - \alpha_i H^i t) \right. \\ & + \sigma_m \frac{e^{\alpha_i H^i t} - e^{\alpha_i H^i (t-1)}}{\alpha_i H^i e^{\beta_i D_t^i}} \\ & \left. + \mu |\log \gamma + \alpha_i H^i t - \beta_i D_t^i - \log \sigma_{\mathcal{C}_i}(t)|^2 \right). \end{aligned} \quad (7.8)$$

We can solve the subproblems using gradient-based methods in a parallel manner. With α and β such learned, γ can be updated using the following equation:

$$\gamma = \exp \left(\frac{\sum_{i=1}^N \sum_{t=1}^{n_i} \log(\sigma_{C_i}(t)/\lambda_{C_i}(t))}{\sum_{i=1}^N n_i} \right). \quad (7.9)$$

Algorithm 8 summarizes our learning algorithm.

Algorithm 8 Learning Proposed Attractiveness Model

- 1: **Input:** Training shots $\{C_i\}_{i=1}^N$, the maximum number of iteration $M = 500$, the parameter $\mu = 0.5$. The gradient descent step size: $\delta_\alpha = 10^{-4}$, $\delta_\beta = 10^{-5}$.
 - 2: **Output:** Parameters of our model α , β , γ .
 - 3: Initialize α^0 , β^0 and γ^0 randomly.
 - 4: **For** $m = 1 : M$
 - 5: **For** $i = 1 : N$
 - 6: $\alpha_i^m = \left(\alpha_i^{m-1} - \delta_\alpha \frac{\partial I_i}{\partial \alpha_i} \Big|_{\alpha_i = \alpha_i^{m-1}} \right)_+$.
 - 7: $\beta_i^m = \left(\beta_i^{m-1} - \delta_\beta \frac{\partial I_i}{\partial \beta_i} \Big|_{\beta_i = \beta_i^{m-1}} \right)_+$.
 - 8: $(\cdot)_+$ sets negative value to be 0.
 - 9: **End**
 - 10: γ^m is calculated by Eq. (7.9).
 - 11: **End**
 - 12: $\alpha = \alpha^m$, $\beta = \beta^m$, $\gamma = \gamma^m$.
-

7.4 Trailer Generation

After learning the attractiveness model from the training set, we are able to generate an attractive trailer given a new testing video \mathcal{V} and a piece of music \mathbf{m} with maximum attractiveness, or equally, minimum loss of attractiveness. The problem of trailer generation can be formulated as:

$$\min_{\mathcal{T} \subset \mathcal{V}} \int_{L_0}^{L_N} \lambda_{\mathcal{T}}(s) ds, \quad (7.10)$$

where $L_0 = 0$ and $L_N = \sum_{i=1}^N n_i$ are the beginning and the ending of the trailer, respectively. n_i is the number of frames in C_i and N is the number of candidate shots selected

from \mathcal{V} . $\{n_i\}$ and N are the positions of the montages. According to Property 3 in Section 7.2.2, they are determined by the saliency points of the music, which is detected from the $\hat{\mathbf{m}}$ in Eq. (7.2). The interval length between adjacent saliency points determines n_i , and the number of saliency points determines N . Since (7.10) is a combinatorial problem and NP-hard, we have to resort to approximate solutions. Inspired by [82], we propose a graph-based method to solve (7.10) approximately and efficiently.

Step 1: Candidate Selection. We rewrite (7.10) as:

$$\min_{\mathcal{T}} \sum_{i=1}^N \int_0^{n_i} \lambda_{\mathcal{C}_i}(s) ds, \quad s.t. \quad \mathcal{T} = \{\mathcal{C}_i \in \mathcal{S}_i\}_{i=1}^N, \quad (7.11)$$

where \mathcal{S}_i , $i = 1, \dots, N$, is the set of s ($s = 5$ in our experiments) candidate shots selected from \mathcal{V} for \mathcal{C}_i . Each selected shot satisfies the following two constraints: 1) the length is not shorter than n_i ; 2) it does not appear in \mathcal{S}_{i-1} . \mathcal{S}_N contains only one shot, which corresponds to the title of the trailer given in advance.

Step 2: Parameter Assignment. For a candidate shot \mathcal{C}_i in the new video, we do not know the parameters of $\lambda_{\mathcal{C}_i}(t)$ in advance. In this paper, we first extract the feature of \mathcal{C}_i as

$$\mathbf{f}_{\mathcal{C}_i} = [H(\hat{f}_1^{(i)}), D(\hat{f}_1^{(i)} || \hat{f}_2^{(i)}), \dots] \in \mathbb{R}^{64}. \quad (7.12)$$

We fix the length of feature vector as 64 in this work — if the length of \mathcal{C}_i is shorter than 64, we pad zeros in the end of $\mathbf{f}_{\mathcal{C}_i}$; if \mathcal{C}_i is longer than 64, we cut the end of $\mathbf{f}_{\mathcal{C}_i}$. Then, we select the matching shot in the training set for \mathcal{C}_i by comparing the features of training shots with those of candidate shots. The matching criterion is the Euclidean distance. The parameters of the matching shot are assigned to $\lambda_{\mathcal{C}_i}(t)$.

Step 3: Graph-based Stitching. (7.11) is still a complicated combinatorial optimization problem. To see this, we note that the selection of \mathcal{C}_{i-1} has recursive influence on the selection of subsequent shots $\{\mathcal{C}_i, \mathcal{C}_{i+1}, \dots\}$. As we know, the initial stimulus in $\lambda_{\mathcal{C}_i}(t)$ is the KL-divergence between the last frame of \mathcal{C}_{i-1} and the first frame of \mathcal{C}_i . Hence, if we

change \mathcal{C}_{i-1} , the intensity function $\lambda_{\mathcal{C}_i}(t)$ will change, and so does the selection of \mathcal{C}_i .

The problem will be efficiently solved if we only consider the pairwise relationships between the shots in the adjacent candidate sets. Given $\{\mathcal{S}_i\}_{i=1}^N$, we can construct a trellis graph \mathcal{G} with $N + 1$ layers. The nodes in the i -th layer are the candidate shots from \mathcal{S}_i . The edge weights in the graph can be defined as follows,

$$w_{p,q}^{i,i+1} = \int_0^{n_i} \lambda_{\mathcal{C}_{p,i}}(s) ds + \int_0^{n_{i+1}} \lambda_{\mathcal{C}_{q,i+1}}(s) ds, \quad (7.13)$$

where $w_{p,q}^{i,i+1}$ is the weight connecting the p -th candidate in \mathcal{S}_i with the q -th candidate in \mathcal{S}_{i+1} . We calculate all the weights independently: the initial stimulus in $\lambda_{\mathcal{C}_{p,i}}(t)$ is the entropy of the first frame of $\mathcal{C}_{p,i}$, which is independent of shot selection in the former layers; on the other hand, $\mathcal{C}_{p,i}$ only influences $\mathcal{C}_{q,i+1}$ through the initial stimulus in $\lambda_{\mathcal{C}_{q,i+1}}$ that is the KL-divergence between the last frame of $\mathcal{C}_{p,i}$ and its first frame. Influence of $\mathcal{C}_{p,i}$ will not propagate to the following layers. In other words, (7.11) can be solved approximately by finding the shortest path [162] in the graph \mathcal{G} (from the first layer to the last one). We summarize our algorithm in Algorithm 9 and illustrate it in Fig. 7.3.

Algorithm 9 Graph-based Trailer Generation Algorithm

- 1: **Input:** a video \mathcal{V} , a piece of music \mathbf{m} , training shots with features and learned parameters.
 - 2: **Output:** a movie trailer \mathcal{T} .
 - 3: Segment \mathcal{V} to shots $\{\mathcal{C}_j\}$ and extract features.
 - 4: For each \mathcal{C}_j , find the matching shot in the training set and assign parameters accordingly.
 - 5: Detect saliency points of \mathbf{m} by (7.2).
 - 6: Construct candidate set \mathcal{S}_i and a hierarchical graph \mathcal{G} .
 - 7: Calculate the weight of edge by (7.5, 7.13).
 - 8: Find the shortest path in \mathcal{G} .
 - 9: \mathcal{T} is constructed by the sequence of shots corresponding to the shortest path associated with the music \mathbf{m} .
-

7.5 Experiments

We conduct two groups of experiments (objective and subjective) to empirically evaluate the proposed method. Our data set consists of 16 publicly available movies including 3 animation movies, 2 fantasy movies, 2 action movies, 5 fiction action movies and 4 dramas in 2012 and 2014. We also collect the movies, their theme music and official trailers. The experimental settings are as follows: we first select 8 of the trailers as the training set and collect the fixation data from 14 volunteers. Then, we learn our attractiveness model as described in Section 7.3.3. Finally, based on the attractiveness model, we produce trailers for the remaining 8 movies following Section 7.4. All movies and their trailers are with frame size 640×480 .

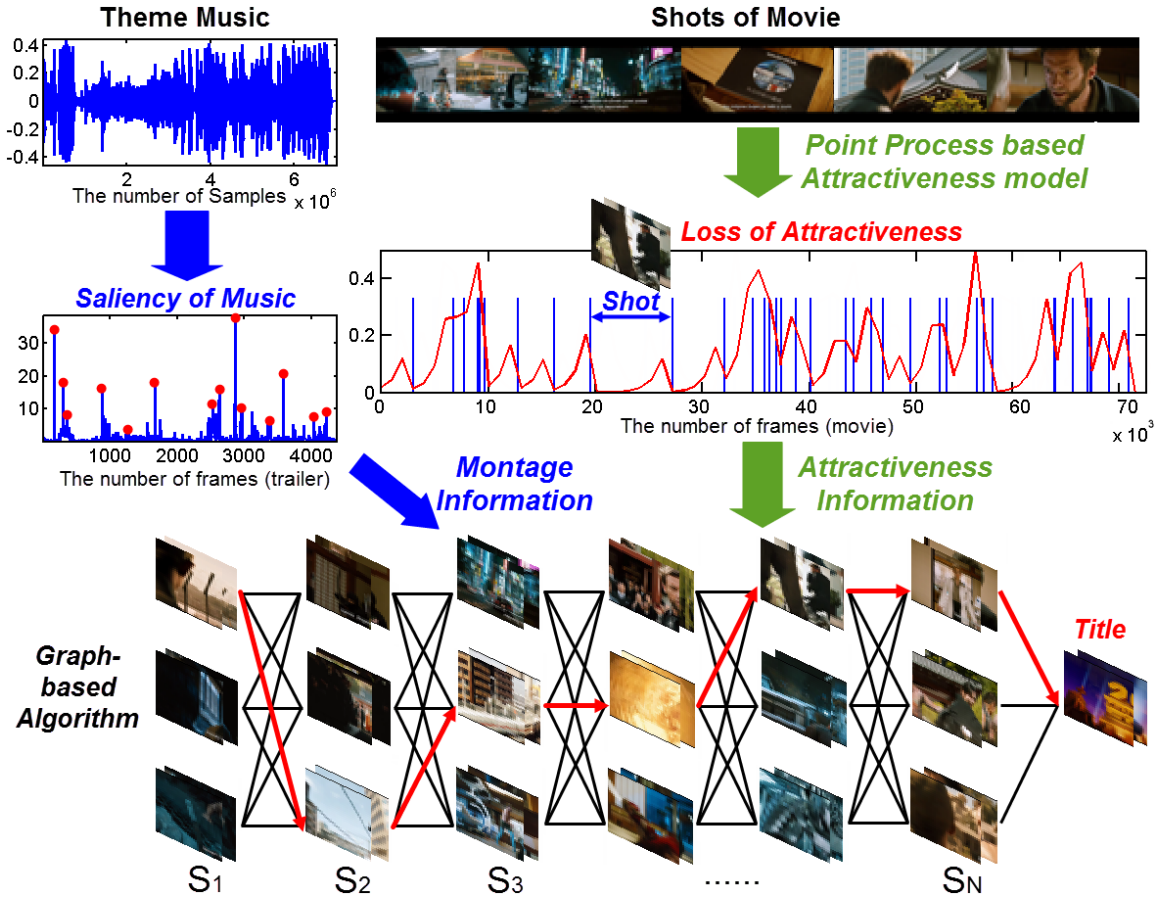


Figure 7.3: The scheme of our trailer generator.

We compare our method (“*Ours*”) with the following four competitors²: *i*) the trailer generator “*V2T*” in [160]; *ii*) the commercial video summarization software “*Muvec*”³; *iii*) the real official trailers (“*RT*”) generated by professionals; *iv*) the real trailers without speech information (“*RTwS*”). For fair comparison, for “*V2T*”, the training and testing sets are the same as we described above; for “*Muvec*”, we generate trailers for the testing movies only as there is no training phase. We also note that, in this work, we focus on the visual attractiveness model and its contribution on trailer generation, and hence we use “*RTwS*” as a baseline. Moreover, we are only able to implement the shot selection and arrangement algorithm of “*V2T*” since other steps such as feature extraction is omitted in the reference.

7.5.1 Objective Evaluation

Loss of Attractiveness. An important criterion for trailers is the loss of attractiveness, which can be approximately measured by the proposed fixation variance. We invite the 14 volunteers to watch the testing trailers generated by all of the five methods mentioned above, and record their fixation points in each frames by an eye tracker. For each method, we calculate the fixation variance σ in each frame for all 8 testing trailers, and hence obtain 32,309 σ ’s. The statistics of σ ’s reflect the overall loss of attractiveness. Specifically, larger σ indicates more loss of attractiveness. We present the mean, the median and the standard deviation of σ ’s for each method in Fig. 7.4.

It is easy to observe that both the mean and the median obtained by our method are the smallest compared with its counterparts. Specifically, the results of our method are comparable to those of “*RT*” whereas the results of “*Muvec*”, “*V2T*” and “*RTwS*” are much higher than those of “*RT*”. Last but not least, the standard deviation results show that the attractiveness of our trailers is the most stable, and “*RT*” trailers are the second best in this aspect. In our opinion, the superiority of our method is mainly based on the utilizing

²Representative trailers generated by all methods are on the website: <https://vimeo.com/user25206850/videos>.

³<http://www.muvec.com/>

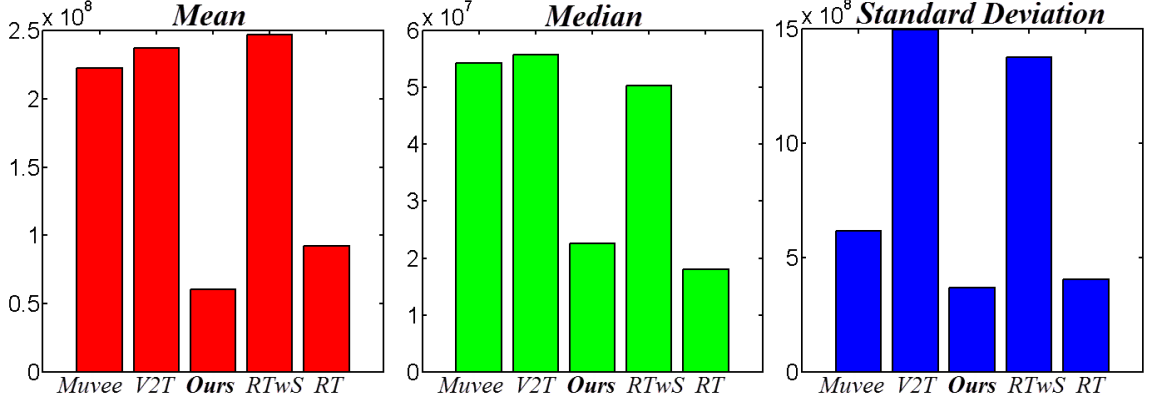


Figure 7.4: The mean, the median and the standard deviation of the fixation variance σ for various methods.

of fixation variance and the proposed point process model. Firstly, fixation variance is real feature from viewers, which reflects the attractiveness of video better than the learned feature from video. Secondly, the proposed point process model captures the dynamics of attractiveness well, which provides us with a useful guidance to generate trailer.

Table 7.2: Comparison on computational cost.

| | Training cost | Testing cost |
|-----------|-------------------------|-------------------------|
| V2T [160] | 0.0024 sec/frame | 0.2676 sec/frame |
| Ours | 0.0014 sec/frame | 0.0113 sec/frame |

Computational Cost. Computation cost is a key factor for trailer generation. As we mentioned in previous sections, it is very promising to have efficient automatic trailer generators that may be potentially applied to millions of online videos. We note that the training and testing computational complexities of “V2T” are both $O(N^3)$, whereas ours are $O(N)$ and $O(N^2)$ for training and testing, respectively.

Table 7.2 compares empirical training and testing cost of our method and “V2T”. Both methods are implemented by MATLAB and run on the same platform (Core i7 CPU @3.40GHz with 32GB memory). Specifically, the training cost is calculated as the model learning time per frame for training set, and the testing cost is calculated as the trailer generation time per frame for the generation result. Table 7.2 validates that our method needs

much less training and testing cost than “V2T”.

7.5.2 Subjective Evaluation

In this subsection, we evaluate our method as well as the baselines through subjective experiments. Similar to [160], for each testing trailer generated by different methods, we invited 14 volunteers to evaluate it by answering the following 3 questions: **Rhythm**: “How well does the montage match with the rhythm of background music?” **Attractiveness**: “How attractive is the trailer?” **Appropriateness**: “How close is the trailer to an real trailer?” For each question, the volunteers were asked to provide an integer score in the range of 1 (lowest) to 7 (highest). Fig. 7.5 shows the overall results for all 8 testing movies.

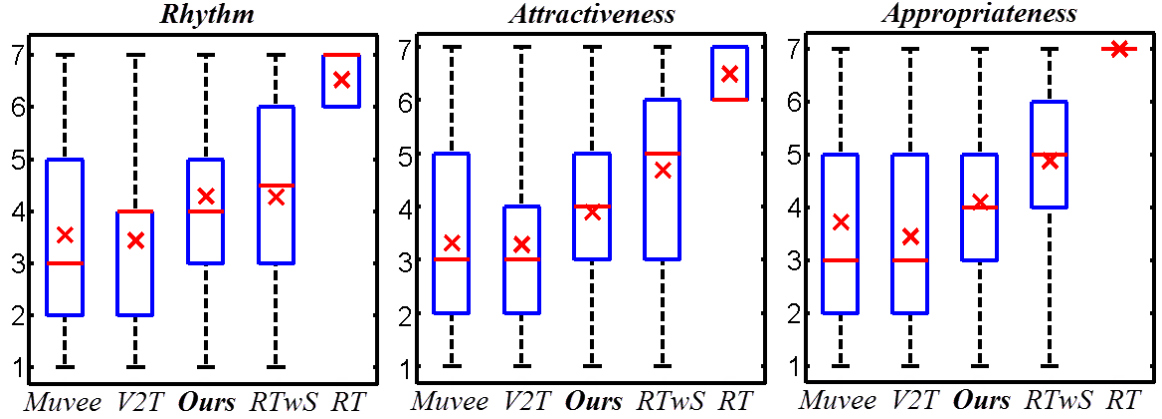


Figure 7.5: The box plots of scores for various methods on three questions. The red crosses are means and the red bars are medians.

Consistency of Objective and Subjective Evaluation. Similar to objective evaluation, we find in Fig. 7.5 that our method is better than “V2T” and “Muvee” in all three questions, indicating that our attractiveness model based on fixation variance and self-correcting point processes is reasonable, and is able to generate trailers that satisfy our subjective feelings. On the other hand, our method are inferior to “RTwS” and “RT”, which is different from the result of objective evaluation. The reasons for the difference may be attributed to that we only use visual information to learn the attractiveness model and produce trailers. On the contrary, official trailers of “RT” often provide speeches, subtitles and special effects

of montages, which impress viewers a lot. Similarly, although the trailers of “*RTwS*” does not have speech information, they still contain subtitles and special effects of montages. Since the information of these factors contribute to raising the attractiveness of the video, volunteers feel that the real trailers are better than our trailers. This observation points out a future extension of our work, that is, in addition to visual and music information, we should also enrich our model with information sources such as speech, subtitles, and montage effects to capture holistic movie attractiveness.

7.6 Summary

In this chapter, we study a challenging problem, namely, automatic trailer generation. To generate attractive trailers, we propose a practical surrogate measure of video attractiveness called fixation variance, and made the first attempt to use point processes to model the attractiveness dynamics. Based upon the attractiveness model, we develop a graph-based music-guided trailer generation method. In the future, we are interested in extending our method to utilize other information such as speeches and subtitles. We would also like to explore parallel algorithms to further improve the scalability of our method.

CHAPTER 8

CONCLUSIONS

In this dissertation, we establish a unified framework for quantitative analysis of asynchronous event sequences, which contains a series of point process-based models and algorithms to deal with practical challenges, e.g., Granger causality analysis for multiple event types, clustering analysis for the event sequences, learning point processes from imperfect observations, and discriminative learning of point processes. These models and algorithms have a wide range of applications, e.g., IPTV system analysis and user clustering, job hopping analysis, constructing dynamical disease networks, ICU patient flow prediction, and video attractiveness modeling. In particular, we have made the following contributions:

- **Causality analysis of event types.** We extend the concept of Granger causality from time series to event sequences and learn Granger causality graph of event types based on Hawkes processes. We show that learning triggering patterns among event types and constructing Granger causality graph are unified in the framework of Hawkes processes. We achieve these two aims simultaneously via learning a nonparametric Hawkes process model combining maximum likelihood estimation with sparse-group-lasso regularization.
- **Clustering analysis of event sequences.** To cluster event sequences effectively, we design a feature-based clustering method and a model-based clustering method, respectively. The feature-based method decomposes the clustering problem into two subproblems: feature extraction and clustering. The feature of each event sequence is first learned in the framework of multi-task learning, and then traditional clustering method is applied to these features. To overcome the drawbacks of the feature-based method, we further establish a Dirichlet mixture model of Hawkes processes, which

has good identifiability and can be learned via variational inference method.

- **Robust learning of point processes.** We enhance the robustness of learning algorithms of point processes in two directions. On one hand, a data synthesis method applying sampling-stitching mechanism is proposed to generate long event sequences from original short doubly-censored ones. This method can be used as pre-processing of learning algorithms, which helps us to learn longtime scale point process models from short term observations. On the other hand, we propose a potential solution to learn point processes in a discriminative way with the help of deep learning techniques. Such a discriminative learning method is very flexible, which suppresses the risk of model misspecification problem.
- **Innovative applications of point processes.** We apply our point process models and learning algorithms to solve many hot issues in the field of data mining, e.g., clustering users based on their behaviors, predicting transition processes of patients among ICUs, constructing dynamical disease networks from patients' admission records. Moreover, we extend the application of point processes to the field of computer vision, proposing an visual attractiveness model based on self-correcting processes. To our knowledge, our work is one of the pioneers utilizing point processes to solve computer vision problems.

In summary, our work contains flexible models and robust learning algorithms for asynchronous event sequences analysis, which has very broad applications covering diverse research fields.

Currently, our work is based on temporal point processes. We believe that it can be extended to spatiotemporal point processes easily and inspire new research directions towards modeling and predicting spatiotemporal event sequences. Additionally, we would like to further enhance the scalability and the flexibility of our methods via introducing more cutting-edge machine learning techniques to our learning algorithms. Therefore, from

the viewpoints of both methodology and application, our future research interests include three points.

- **Scalable learning of point processes.** In practical applications, hundreds and thousands of event sequences will be involved in the learning phase. A scalable algorithm is required to learn point process models from big data. To achieve this aim, we plan to investigate how to accelerate the learning of point processes and reduce the time and space complexity of our algorithms. The approximate learning method will be developed, and the techniques of parallel computing, distributed learning, and online learning will be introduced to our learning algorithms.
- **Deep combination of deep learning and point processes.** In recent years, deep learning has been widely used to learn representative features from chaotic and complicated data. This technique can also be combined with point processes, as we mentioned in Chapter 6. However, the combination proposed in this dissertation is very simple and there might be more sophisticated solutions to introduce deep learning into point process models. Along this direction, we plan to design a hierarchical point process model to imitate the architecture of deep neural networks.
- **E-health based on Point processes.** The diagnose and the treatment of a disease are often coupled with each other. The relationships between diagnoses and treatments may be mutually-exciting (i.e., a treatment is applied after a diagnose and after a period another diagnose would be applied to evaluate the treatment), while the relationship among diagnoses and that among treatments are self-correcting (i.e., multiple treatments are unlikely to be applied simultaneously without diagnose). We expect to capture the complicated relationship between diagnoses and treatments via a pair of interactive point processes and achieve personalized treatment recommendation given patients' historical diagnoses and treatments.

Appendices

APPENDIX A

DETAILS OF BASIS FUNCTION SELECTION

In our model, the intensity function of time-invariant Hawkes process over all dimensions is:

$$\begin{aligned}
\lambda(t) &= \sum_{c=1}^C \lambda_c(t) \\
&= \sum_{c=1}^C \left(\mu_c + \sum_{c'=1}^C \int_0^t \phi_{cc'}(s) dN_{c'}(t-s) \right) \\
&= \sum_{c=1}^C \mu_c + \sum_{c=1}^C \sum_{t_i < t} \phi_{cc_i}(t-t_i) \\
&= \sum_{c=1}^C \mu_c + \sum_{c=1}^C \sum_{t_i < t} \sum_{m=1}^M a_{cc_im} \kappa_m(t-t_i).
\end{aligned} \tag{A.1}$$

Applying Fourier transform, we have

$$\begin{aligned}
\hat{\lambda}(\omega) &= \sum_{c=1}^C \mu_c \sqrt{2\pi} \delta(\omega) \\
&\quad + \sum_{c=1}^C \sum_{t_i < t} \sum_{m=1}^M a_{cc_im} e^{-j\omega t_i} \hat{\kappa}_m(\omega).
\end{aligned} \tag{A.2}$$

In other words, the spectral of $\lambda(t)$ is the weighted sum of those of basis functions. Therefore, the cut-off frequency of basis function is bounded by that of intensity function.

Given training sequences $\mathcal{S} = \{\mathbf{s}_n\}_{n=1}^N$, where $\mathbf{s}_n = \{(t_i^n, c_i^n)\}_{i=1}^{I_n}$, we can estimate $\lambda(t)$ empirically via a Gaussian-based kernel density estimator:

$$\lambda(t) = \sum_{n=1}^N \sum_{i=1}^{I_n} G_h(t-t_i^n). \tag{A.3}$$

Here t_i^n is the time stamp of the i -th event at the n -th sequence. $G_h(t-t_i^n) = \exp(-\frac{(t-t_i^n)^2}{2h^2})$

is a Gaussian kernel with the bandwidth h .

Because we only care about the selection of basis functions, we just need to estimate the spectral of $\lambda(t)$ rather than compute (2.15) directly. Specifically, applying Silverman's rule of thumb [86], we first set optimal $h = (\frac{4\hat{\sigma}^5}{3\sum_c I_n})^{0.2}$, where $\hat{\sigma}$ is the standard deviation of time stamps $\{t_i^n\}$. Applying Fourier transform, we compute an upper bound for the spectral of $\lambda(t)$ as

$$\begin{aligned}
|\hat{\lambda}(\omega)| &= \left| \int_{-\infty}^{\infty} \lambda(t) e^{-j\omega t} dt \right| \\
&= \left| \sum_{n=1}^N \sum_{i=1}^{I_n} \int_{-\infty}^{\infty} e^{-\frac{(t-t_i^n)^2}{2h^2}} e^{-j\omega t} dt \right| \\
&\leq \sum_{n=1}^N \sum_{i=1}^{I_n} \left| \int_{-\infty}^{\infty} e^{-\frac{(t-t_i^n)^2}{2h^2}} e^{-j\omega t} dt \right| \\
&= \sum_{n=1}^N \sum_{i=1}^{I_n} \left| e^{-j\omega t_i^n} e^{-\frac{\omega^2 h^2}{2}} \sqrt{2\pi h^2} \right| \\
&\leq \sum_{n=1}^N \sum_{i=1}^{I_n} \left| e^{-j\omega t_i^n} \right| \left| e^{-\frac{\omega^2 h^2}{2}} \sqrt{2\pi h^2} \right| \\
&= \left(\sum_{n=1}^N I_n \sqrt{2\pi h^2} \right) e^{-\frac{\omega^2 h^2}{2}}.
\end{aligned} \tag{A.4}$$

Furthermore, we can compute the upper bound of the absolute sum of the spectral

higher than ω_0 as

$$\begin{aligned}
& \int_{\omega_0}^{\infty} |\hat{\lambda}(\omega)| d\omega \\
& \leq \left(\sum_{n=1}^N I_n \sqrt{2\pi h^2} \right) \int_{\omega_0}^{\infty} e^{-\frac{\omega^2 h^2}{2}} d\omega \\
& = 2\pi \left(\sum_{n=1}^N I_n \right) \int_{\omega_0}^{\infty} \frac{h}{\sqrt{2\pi}} e^{-\frac{\omega^2 h^2}{2}} d\omega \\
& = 2\pi \left(\sum_{n=1}^N I_n \right) \left(\frac{1}{2} - \int_0^{\omega_0} \frac{h}{\sqrt{2\pi}} e^{-\frac{\omega^2 h^2}{2}} d\omega \right) \\
& = 2\pi \left(\sum_{n=1}^N I_n \right) \left(\frac{1}{2} - \frac{1}{2} \int_{-\omega_0}^{\omega_0} \frac{h}{\sqrt{2\pi}} e^{-\frac{\omega^2 h^2}{2}} d\omega \right) \\
& = \pi \left(\sum_{n=1}^N I_n \right) \left(1 - \frac{1}{\sqrt{2}} \text{erf}(\omega_0 h) \right),
\end{aligned} \tag{A.5}$$

where $\text{erf}(x) = \frac{1}{\sqrt{\pi}} \int_{-x}^x e^{-t^2} dt$.

Therefore, give a bound of residual ϵ , we can find an ω_0 guaranteeing $\int_{\omega_0}^{\infty} |\hat{\lambda}(\omega)| d\omega \leq \epsilon$, or $\text{erf}(\omega_0 h) \geq \sqrt{2} - \frac{\sqrt{2}\epsilon}{\pi \sum_{n=1}^N I_n}$. The proposed basis functions $\{\kappa_{\omega_0}(t, t_m)\}_{m=1}^M$ are selected, where ω_0 is the cut-off frequency of basis function and $t_m = \frac{(m-1)T}{M}$, $M = \lceil \frac{T\omega_0}{\pi} \rceil$.

APPENDIX B

IMPLEMENTATION DETAILS OF DMHP

B.1 Nested EM Framework

We consider a variational distribution having the following factorization:

$$q(\mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\mu}, \mathbf{A}) = q(\mathbf{Z})q(\boldsymbol{\pi}, \boldsymbol{\mu}, \mathbf{A}) = q(\mathbf{Z})q(\boldsymbol{\pi}) \prod_k q(\boldsymbol{\mu}^k)q(\mathbf{A}^k). \quad (\text{B.1})$$

An nested EM algorithm can be used to optimize (B.1).

Update Responsibility (E-step). In each *outer iteration*, the logarithm of the optimized factor $q^*(\mathbf{Z})$ is approximated as

$$\begin{aligned} & \log q^*(\mathbf{Z}) \\ &= \mathbb{E}_{\boldsymbol{\pi}, \boldsymbol{\mu}, \mathbf{A}} [\log p(\mathbf{S}, \mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\mu}, \mathbf{A})] + \text{C} \\ &= \mathbb{E}_{\boldsymbol{\pi}} [\log p(\mathbf{Z}|\boldsymbol{\pi})] + \mathbb{E}_{\boldsymbol{\mu}, \mathbf{A}} [\log p(\mathbf{S}|\mathbf{Z}, \boldsymbol{\mu}, \mathbf{A})] + \text{C} \\ &= \sum_{n,k} z_{nk} (\mathbb{E}[\log \pi^k] + \mathbb{E}[\log \text{HP}(\mathbf{s}_n|\boldsymbol{\mu}^k, \mathbf{A}^k)]) + \text{C} \\ &= \sum_{n,k} z_{nk} \left(\mathbb{E}[\log \pi^k] + \mathbb{E} \left[\sum_i \log \lambda_{c_i}^k(t_i) - \sum_c \int_0^{T_n} \lambda_c^k(s) ds \right] \right) + \text{C} \quad (\text{B.2}) \\ &\approx \sum_{n,k} z_{nk} \left(\mathbb{E}[\log \pi^k] + \sum_i \left(\log \mathbb{E}[\lambda_{c_i}^k(t_i)] - \frac{\text{Var}[\lambda_{c_i}^k(t_i)]}{2\mathbb{E}^2[\lambda_{c_i}^k(t_i)]} \right) \right. \\ &\quad \left. - \sum_c \mathbb{E} \left[\int_0^{T_n} \lambda_c^k(s) ds \right] \right) + \text{C} \\ &= \sum_{n,k} z_{nk} \log \rho_{nk} + \text{C}. \end{aligned}$$

where C is a constant, and each term $\mathbb{E}[\log \lambda_c^k(t)]$ is approximated via its second-order

Taylor expansion $\log \mathbb{E}[\lambda_c^k(t)] - \frac{\text{Var}[\lambda_c^k(t)]}{2\mathbb{E}^2[\lambda_c^k(t)]}$ [97]. Then, we have

$$\begin{aligned}
& \log \rho_{nk} \\
&= \mathbb{E}[\log \pi^k] + \sum_i \left(\log(\mathbb{E}[\lambda_{c_i}^k(t_i)]) - \frac{\text{Var}[\lambda_{c_i}^k(t_i)]}{2\mathbb{E}^2[\lambda_{c_i}^k(t_i)]} \right) - \sum_c \mathbb{E} \left[\int_0^{T_n} \lambda_c^k(s) ds \right] \\
&= \mathbb{E}[\log \pi^k] + \sum_i \left(\log(\mathbb{E}[\mu_{c_i}^k]) + \sum_{j < i, d} \mathbb{E}[a_{c_i c_j d}^k g_d(\tau_{ij})] - \frac{\text{Var}[\mu_{c_i}^k] + \sum_{j < i, d} \text{Var}[a_{c_i c_j d}^k g_d^2(\tau_{ij})]}{2(\mathbb{E}[\mu_{c_i}^k] + \sum_{j < i, d} \mathbb{E}[a_{c_i c_j d}^k g_d(\tau_{ij})])^2} \right) \\
&\quad - \sum_c (T_n \mathbb{E}[\mu_c^k] + \sum_{i, d} \mathbb{E}[a_{cc_i d}^k G_d(T_n - t_i)]) \\
&= \mathbb{E}[\log \pi^k] + \sum_i \left(\log \left(\sqrt{\frac{\pi}{2}} \beta_{c_i}^k + \sum_{j < i, d} \sigma_{c_i c_j d}^k g_d(\tau_{ij}) \right) - \frac{\frac{4-\pi}{2} (\beta_{c_i}^k)^2 + \sum_{j < i, d} (\sigma_{c_i c_j d}^k g_d(\tau_{ij}))^2}{2(\sqrt{\frac{\pi}{2}} \beta_{c_i}^k + \sum_{j < i, d} \sigma_{c_i c_j d}^k g_d(\tau_{ij}))^2} \right) \\
&\quad - \sum_c (T_n \sqrt{\frac{\pi}{2}} \beta_c^k + \sum_{i, d} \sigma_{cc_i d}^k G_d(T_n - t_i)),
\end{aligned}$$

where $G_d(t) = \int_0^t g_d(s) ds$ and $\tau_{ij} = t_i - t_j$. The second equation above is based on the prior that all of the parameters are independent to each other. The term $\mathbb{E}[\log \pi^k] = \psi(\alpha_k) - \psi(\sum_k \alpha_k)$, where $\psi(\cdot)$ is the digamma function.¹ Then, the responsibility r_{nk} is calculated as

$$r_{nk} = \mathbb{E}[z_{nk}] = \frac{\rho_{nk}}{\sum_j \rho_{nj}}, \text{ and } N_k = \sum_n r_{nk}. \quad (\text{B.3})$$

It should be noted that here we increase $q^*(\mathbf{Z})$ via maximizing its upper bound in each iteration because the difference between $q^*(\mathbf{Z})$ and its upper bound is bounded tightly. In particular, $q^*(\mathbf{Z})$ in (B.2) involves $\mathbb{E}[\log \lambda_{c_i}^k(t_i)]$, which is approximated via Jensen's inequality as $\log \mathbb{E}[\lambda_{c_i}^k(t_i)]$. It actually is the first order Talyor expansion of $\mathbb{E}[\log \lambda_{c_i}^k(t_i)]$. The second order term is bounded well and the higher order terms can be ignored. We prove the rationality of our relaxation in the appendix.

¹Denote the gamma function as $\Gamma(t) = \int_0^\infty x^{t-1} e^{-x} dx$, the digamma function is defined as $\psi(t) = \frac{d}{dt} \ln \Gamma(t)$.

Update Parameters (M-step). The optimal factor $q^*(\boldsymbol{\pi}, \boldsymbol{\mu}, \mathbf{A})$ is

$$\begin{aligned} & \log q^*(\boldsymbol{\pi}, \boldsymbol{\mu}, \mathbf{A}) \\ &= \sum_k \log(p(\boldsymbol{\mu}^k)p(\mathbf{A}^k)) + \mathbb{E}_{\mathbf{Z}}[\log p(\mathbf{Z}|\boldsymbol{\pi})] \\ & \quad + \log p(\boldsymbol{\pi}) + \sum_{n,k} r_{nk} \log \text{HP}(\mathbf{s}_n|\boldsymbol{\mu}^k, \mathbf{A}^k) + \text{C}. \end{aligned} \quad (\text{B.4})$$

We can estimate the parameters of Hawkes processes via:

$$\max_{\boldsymbol{\mu}, \mathbf{A}} \log(p(\boldsymbol{\mu})p(\mathbf{A})) + \sum_{n,k} r_{nk} \log \text{HP}(\mathbf{s}_n|\boldsymbol{\mu}^k, \mathbf{A}^k).$$

Here, we need to use an iterative method to solve the above optimization problem. Specifically, we initialize $\boldsymbol{\mu}$ and \mathbf{A} via the expectations of their distributions (used in E-step), i.e., $\boldsymbol{\mu} = \sqrt{\frac{\pi}{2}}\mathbf{B}$ and $\mathbf{A} = \boldsymbol{\Sigma}$. Applying the Jensen's inequality, we obtain the surrogate function of the objective function:

$$\begin{aligned} & \log(p(\boldsymbol{\mu})p(\mathbf{A})) + \sum_{n,k} r_{nk} \log \text{HP}(\mathbf{s}_n|\boldsymbol{\mu}^k, \mathbf{A}^k) \\ &= \sum_{c,k} \left[\log \mu_c^k - \frac{1}{2} \left(\frac{\mu_c^k}{\beta_c^k} \right)^2 \right] - \sum_{c,c',d,k} \frac{a_{cc'd}^k}{\sigma_{cc'd}^k} + \sum_{n,k} r_{nk} \left[\sum_i \log \lambda_{c_i}^k(t_i) - \sum_c \int_0^{T_n} \lambda_c^k(s) ds \right] \\ &\geq \sum_{c,k} \left[\log \mu_c^k - \frac{1}{2} \left(\frac{\mu_c^k}{\beta_c^k} \right)^2 \right] - \sum_{c,c',d,k} \frac{a_{cc'd}^k}{\sigma_{cc'd}^k} + \sum_{n,k} r_{nk} \left[\sum_i \left(p_{ii}^k \log \frac{\mu_{c_i}^k}{p_{ii}^k} \right. \right. \\ & \quad \left. \left. + \sum_{j < i, d} p_{ij d}^k \log \frac{a_{c_i c_j d}^k g_d(\tau_{ij})}{p_{ij d}^k} \right) - \sum_c T_n \mu_c^k - \sum_{c,i,d} a_{cc_i d}^k G_d(T_n - t_i) \right] \\ &= Q, \end{aligned}$$

where $p_{ii}^k = \frac{\mu_{c_i}^k}{\lambda_{c_i}^k(t_i)}$, and $p_{ij d}^k = \frac{a_{c_i c_j d}^k g_d(\tau_{ij})}{\lambda_{c_i}^k(t_i)}$. Setting $\frac{\partial Q}{\partial \mu_c^k} = 0$ and $\frac{\partial Q}{\partial a_{cc'd}^k} = 0$, we have

$$\hat{\mu}_c^k = \frac{-b + \sqrt{b^2 - 4ac}}{2a}, \quad \hat{a}_{cc'd}^k = \frac{\sum_n r_{nk} \sum_{i:c_i=c} \sum_{j:c_j=c'} p_{ij d}^k}{1/\sigma_{cc'd}^k + \sum_n r_{nk} \sum_{i:c_i=c'} G_d(T_n - t_i)}. \quad (\text{B.5})$$

where $a = \frac{1}{(\beta_c^k)^2}$, $b = \sum_n r_{nk} T_n$, $c = -1 - \sum_n r_{nk} \sum_{i:c_i=c} p_{ii}^k$. After repeating several such *inner iterations*, we can get optimal $\hat{\boldsymbol{\mu}}$, $\hat{\mathbf{A}}$, and update distributions as

$$\boldsymbol{\Sigma}^k = \hat{\mathbf{A}}^k, \mathbf{B}^k = \sqrt{2/\pi} \hat{\boldsymbol{\mu}}^k. \quad (\text{B.6})$$

The distribution of clusters can be estimated via $\pi^k = \frac{N_k}{N}$.

B.2 Update The Number of Clusters K via MCMC

When the number of clusters K is unknown, we can apply the Markov chain Monte Carlo (MCMC) [98, 99, 163] to estimate K via merging or splitting clusters in the learning phase.

Chose move type. We make a random choice to propose a combine or a split move. Let q_m and $q_s = 1 - q_m$ denote the probability of proposing a merge and a split move, respectively, for a current K . Following the work in [163], we use $q_m = 0.5$ for $K \geq 2$, and $q_m = 0$ for $K = 1$.

Merge move. We randomly select a pair (k_1, k_2) of components to merge and form a new component k . The probability of choosing (k_1, k_2) is $q_c(k_1, k_2) = \frac{1}{K(K-1)}$. For our model, we can apply the following deterministic transformation to get new merged parameters:

$$\pi^k = \pi^{k_1} + \pi^{k_2}, \quad \mathbf{A}^k = \frac{\pi^{k_1}}{\pi^k} \mathbf{A}^{k_1} + \frac{\pi^{k_2}}{\pi^k} \mathbf{A}^{k_2}, \quad \boldsymbol{\mu}^k = \frac{\pi^{k_1}}{\pi^k} \boldsymbol{\mu}^{k_1} + \frac{\pi^{k_2}}{\pi^k} \boldsymbol{\mu}^{k_2}. \quad (\text{B.7})$$

Then $\boldsymbol{\Sigma}$ and \mathbf{B} are updated accordingly.

Split move. We randomly select a component k to split into two new components k_1 and k_2 . The probability of choosing component k is $q_s(k) = \frac{1}{K}$. Different from the sampling method in previous work [98, 99, 163], the splitting of parameters is an ill-posed problem with positive constraints. Here, we apply a simple heuristic transformation to get

new splitting parameters:

$$\begin{aligned} \pi^{k_1} &= a\pi^k, \pi^{k_2} = (1-a)\pi^k, a \sim Be(1, 1), \\ \mathbf{A}^{k_1} &= \frac{1}{2a}\mathbf{A}^k, \mathbf{A}^{k_2} = \frac{1}{2(1-a)}\mathbf{A}^k, \quad \boldsymbol{\mu}^{k_1} = \frac{1}{2a}\boldsymbol{\mu}^k, \boldsymbol{\mu}^{k_2} = \frac{1}{2(1-a)}\boldsymbol{\mu}^k. \end{aligned} \tag{B.8}$$

Then Σ and B are updated accordingly.

Acceptance. Given original parameters Θ and the new Θ' , we accept a merge/split move with the probability $\min\{1, \text{likelihood ratio} \times \frac{p(\Theta')}{p(\Theta)}\}$.

APPENDIX C

THE PROOF OF LOCAL IDENTIFIABILITY OF DMHP

Before proving the local identifiability of our DMHP model, we first introduce some key concepts. A temporal point process is a random process whose realization consists of a list of discrete events in time $\{t_i\}$ with $t_i \in [0, T]$. Here $[0, T]$ is the time interval of the process. It can be equivalently represented as a counting process, $N = \{N(t) | t \in [0, T]\}$, where $N(t)$ records the number of events before time t . A multi-dimensional point process with C types of event is represented by C counting processes $\{N_c\}_{c=1}^C$ on a probability space $(\Omega, \mathfrak{F}, \mathbb{P})$. $N_c = \{N_c(t) | t \in [0, T]\}$, where $N_c(t)$ is the number of type- c events occurring at or before time t . $\Omega = [0, T] \times \mathcal{C}$ is the sample space. $\mathcal{C} = \{1, \dots, C\}$ is the set of event types. $\mathfrak{F} = (\mathfrak{F}(t))_{t \in \mathbb{R}}$ is the filtration representing the set of events sequence the process can realize until time t . \mathbb{P} is the probability measure.

Hawkes process is a kind of temporal point processes having self-and mutually-triggering patterns. The triggering of historical events on current ones in a Hawkes process can be modeled as branch processes [164, 32]. As a result, Hawkes Process can be represented as a superposition of many non-homogeneous Poisson process. Due to the superposition theorem of Poisson processes, the superposition of the individual processes is equivalent to the point process with summation of their intensity function. Given this we can break the counting process associated to each addition to the intensity function (or associated to each event): $N(t) = \sum_{i=0}^n N^i(t)$, where $N^0(t)$ is the counting process associated to the baseline intensity $\mu(t)$ and $N^i(t)$ is the non-homogenous Poisson process for the i -th branch. Similarly, we can write the intensity function of Hawkes process as $\lambda(t) = \sum_{i=0}^n \lambda^i(t)$, where $\lambda^i(t)$ is the intensity of the i -th branch.

Definition C.0.1. Two parameter points Θ^1 and Θ^2 are said to be observationally equivalent if $p(s; \Theta^1) = p(s; \Theta^2)$ for all samples s 's in sample space.

Definition C.0.2. A parameter point Θ^0 is said to be locally identifiable if there exists an open neighborhood of Θ^0 containing no other Θ in the parameter space which is observationally equivalent.

Definition C.0.3. Let $I(\Theta)$ be a matrix whose elements are continuous functions of Θ everywhere in the parameter space. The point Θ^0 is said to be a regular point of the matrix if there exists an open neighborhood of Θ^0 in which $I(\Theta)$ has constant rank.

The information matrix $I(\Theta)$ is defined as

$$\begin{aligned} I(\Theta) &= \mathbb{E}_s \left[\frac{\partial \log p(s; \Theta)}{\partial \Theta} \frac{\partial \log p(s; \Theta)}{\partial \Theta^\top} \right] \\ &= \mathbb{E}_s \left[\frac{1}{p^2(s; \Theta)} \frac{\partial p(s; \Theta)}{\partial \Theta} \frac{\partial p(s; \Theta)}{\partial \Theta^\top} \right], \end{aligned}$$

The local identifiability of our DMHP model is based on the following two theorems.

Theorem C.0.1. [104] The information matrix $I(\Theta)$ is positive definite if and only if there does not exist a nonzero vector of constants w such that $w^\top \frac{\partial p(s; \Theta)}{\partial \Theta} = 0$ for all samples s 's in sample space.

Theorem C.0.2. [105] Let Θ^0 be a regular point of the information matrix $I(\Theta)$. Then Θ^0 is locally identifiable if and only if $I(\Theta^0)$ is nonsingular.

To our DMHP model, the log-likelihood function is composed with differentiable functions of Θ . Therefore, the elements of information matrix $I(\Theta)$ are continuous functions w.r.t. Θ in the parameter space. According to Theorems C.0.1 and C.0.2, our Theorem 4.3.1 holds if and only if to each vector $\frac{\partial p(s; \Theta)}{\partial \Theta}$ w.r.t. a point Θ , there does not exist a nonzero vector of constants w such that $w^\top \frac{\partial p(s; \Theta)}{\partial \Theta} = 0$ for all event sequences $s \in \mathfrak{F}$.

Assume that there exists a nonzero w such that $w^\top \frac{\partial p(s; \Theta)}{\partial \Theta} = 0$ for all $s \in \mathfrak{F}$. We have the following **counter-evidence**: Considering the simplest case — the mixture of two Poisson processes (or equivalently, two 1-dimensional Hawkes processes whose impact

functions $\phi(t) \equiv 0$), we can write its likelihood given a sequence with N events in $[0, T]$ as

$$\begin{aligned} p(\mathbf{s}_N; \Theta) &= \pi \lambda_1^N \exp(-T\lambda_1) + (1 - \pi) \lambda_2^N \exp(-T\lambda_2) \\ &= \Lambda_1 + \Lambda_2, \end{aligned}$$

where $\Theta = [\pi, \lambda_1, \lambda_2]^\top$, $\lambda_1 \neq \lambda_2$. According to our assumption, we have

$$\mathbf{w}^\top \frac{\partial p(\mathbf{s}_N; \Theta)}{\partial \Theta} = \mathbf{w}^\top \begin{bmatrix} \frac{\Lambda_1}{\pi} - \frac{\Lambda_2}{1-\pi} \\ (\frac{N}{\lambda_1} - T)\Lambda_1 \\ (\frac{N}{\lambda_2} - T)\Lambda_2 \end{bmatrix} = 0,$$

Denote the time stamp of the last event as t_N , we can generate new event sequences $\{\mathbf{s}_{N+n}\}_{n=1}^\infty$ via adding n events in $(t_N, T]$, and

$$\mathbf{w}^\top \frac{\partial p(\mathbf{s}_{N+n}; \Theta)}{\partial \Theta} = \mathbf{w}^\top \begin{bmatrix} \lambda_1^n \frac{\Lambda_1}{\pi} - \lambda_2^n \frac{\Lambda_2}{1-\pi} \\ ((N+n) - T\lambda_1)\lambda_1^{n-1}\Lambda_1 \\ ((N+n) - T\lambda_2)\lambda_2^{n-1}\Lambda_2 \end{bmatrix}.$$

$\mathbf{w}^\top \frac{\partial p(\mathbf{s}_{N+n}; \Theta)}{\partial \Theta} = 0$ for $n = 0, \dots, \infty$ requires $\mathbf{w} \equiv \mathbf{0}$ or all $\frac{\partial p(\mathbf{s}_{N+n}; \Theta)}{\partial \Theta}$ are coplanar. However, according to the formulation above, for arbitrary three different $n_1, n_2, n_3 \in \{0, \dots, \infty\}$, $\sum_{i=1}^3 \alpha_i \frac{\partial p(\mathbf{s}_{N+n_i}; \Theta)}{\partial \Theta} = \mathbf{0}$ holds if and only if $\alpha_1 = \alpha_2 = \alpha_3 = 0$.¹ Therefore, $\mathbf{w} \equiv \mathbf{0}$, which violates the assumption above.

Such a counter-evidence can also be found in more general case, i.e., mixtures of multiple multi-dimensional Hawkes processes because Hawkes process is a superposition of many non-homogeneous Poisson process. As a result, according to Theorems C.0.1 and C.0.2, each point Θ in the parameter space is regular point of $I(\Theta)$ and the $I(\Theta)$ is nonsingular, and thus, our DMHP model is locally identifiable.

¹The derivation is simple. Interested reader can try the case with $n_1 = 0, n_2 = 1, n_3 = 3$

REFERENCES

- [1] D. J. Daley and D. Vere-Jones, *An introduction to the theory of point processes: Volume II: General theory and structure*. Springer Science & Business Media, 2007.
- [2] V. Didelez, “Graphical models for marked point processes based on local independence,” *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 70, no. 1, pp. 245–264, 2008.
- [3] A. Arnold, Y. Liu, and N. Abe, “Temporal causal modeling with graphical granger methods,” in *SIGKDD*, 2007.
- [4] M. Eichler, “Graphical modelling of multivariate time series,” *Probability Theory and Related Fields*, vol. 153, no. 1-2, pp. 233–268, 2012.
- [5] F. Han and H. Liu, “Transition matrix estimation in high dimensional time series,” in *ICML*, 2013.
- [6] S. Basu, A. Shojaie, and G. Michailidis, “Network granger causality with inherent grouping structure,” *Journal of Machine Learning Research*, vol. 16, pp. 417–453, 2015.
- [7] W. Lian, R. Henao, V. Rao, J. Lucas, and L. Carin, “A multitask point process predictive model,” in *ICML*, 2015.
- [8] J. J. Van Wijk and E. R. Van Selow, “Cluster and calendar based visualization of time series data,” in *IEEE Symposium on Information Visualization*, 1999.
- [9] E. A. Maharaj, “Cluster of time series,” *Journal of Classification*, vol. 17, no. 2, pp. 297–314, 2000.
- [10] T. W. Liao, “Clustering of time series data: A survey,” *Pattern recognition*, vol. 38, no. 11, pp. 1857–1874, 2005.
- [11] B. F. Cole, M. Bonetti, A. M. Zaslavsky, and R. D. Gelber, “A multistate markov chain model for longitudinal, categorical quality-of-life data subject to non-ignorable missingness,” *Statistics in Medicine*, vol. 24, no. 15, pp. 2317–2334, 2005.
- [12] C. Meek, “Toward learning graphical and causal process models,” in *UAI Workshop Causal Inference: Learning and Prediction*, 2014.

- [13] L. R. Rabiner and B.-H. Juang, “An introduction to hidden markov models,” *ASSP Magazine, IEEE*, vol. 3, no. 1, pp. 4–16, 1986.
- [14] B. Cooper and M. Lipsitch, “The analysis of hospital infection data using hidden markov models,” *Biostatistics*, vol. 5, no. 2, pp. 223–237, 2004.
- [15] F. Ginter, H. Suominen, S. Pyysalo, and T. Salakoski, “Combining hidden markov models and latent semantic analysis for topic segmentation and labeling: Method and clinical application,” *International Journal of Medical Informatics*, vol. 78, no. 12, e1–e6, 2009.
- [16] M. Iannelli and A. Pugliese, “Continuous-time markov chains,” in *An Introduction to Mathematical Population Dynamics*, Springer, 2014, pp. 329–334.
- [17] A. Król and P. Saint-Pierre, “Semimarkov: An r package for parametric estimation in multi-state semi-markov models,” *Journal of Statistical Software*, vol. 66, no. 1, pp. 1–16, 2015.
- [18] Y.-Y. Liu, S. Li, F. Li, L. Song, and J. M. Rehg, “Efficient learning of continuous-time hidden markov models for disease progression,” in *NIPS*, 2015.
- [19] S.-H. Yang and H. Zha, “Mixture of mutually exciting processes for viral diffusion,” in *ICML*, 2013.
- [20] L. Li and H. Zha, “Learning parametric models for social infectivity in multi-dimensional hawkes processes,” in *AAAI*, 2014.
- [21] Q. Zhao, M. A. Erdogdu, H. Y. He, A. Rajaraman, and J. Leskovec, “Seismic: A self-exciting point process model for predicting tweet popularity,” in *SIGKDD*, 2015.
- [22] D. Luo, H. Xu, Y. Zhen, X. Ning, H. Zha, X. Yang, and W. Zhang, “Multi-task multi-dimensional hawkes processes for modeling event sequences,” in *IJCAI*, 2015.
- [23] J. Yan, C. Zhang, H. Zha, M. Gong, C. Sun, J. Huang, S. Chu, and X. Yang, “On machine learning towards predictive sales pipeline analytics,” in *AAAI*, 2015.
- [24] V. Isham and M. Westcott, “A self-correcting point process,” *Stochastic Processes and Their Applications*, vol. 8, no. 3, pp. 335–347, 1979.
- [25] Y. Ogata and D. Vere-Jones, “Inference for earthquake models: A self-correcting model,” *Stochastic processes and their applications*, vol. 17, no. 2, pp. 337–347, 1984.

- [26] H. Xu, Y. Zhen, and H. Zha, “Trailer generation via a point process-based visual attractiveness model,” in *IJCAI*, 2015.
- [27] Y. Zhao, X. Qi, Z. Liu, Y. Zhang, and T. Zheng, “Mining medical records with a klipi multi-dimensional hawkes model,” in *KDD Workshop on Health Informatics*, 2015.
- [28] A. G. Hawkes and D. Oakes, “A cluster process representation of a self-exciting process,” *Journal of Applied Probability*, pp. 493–503, 1974.
- [29] H. Xu, M. Farajtabar, and H. Zha, “Learning granger causality for hawkes processes,” in *ICML*, 2016.
- [30] K. Zhou, H. Zha, and L. Song, “Learning triggering kernels for multi-dimensional hawkes processes,” in *ICML*, 2013.
- [31] E. Bacry, S. Delattre, M. Hoffmann, and J.-F. Muzy, “Some limit theorems for hawkes processes and application to financial statistics,” *Stochastic Processes and their Applications*, vol. 123, no. 7, pp. 2475–2499, 2013.
- [32] M. Farajtabar, N. Du, M. Gomez-Rodriguez, I. Valera, H. Zha, and L. Song, “Shaping social activity by incentivizing users,” in *NIPS*, 2014.
- [33] J. G. Rasmussen, “Bayesian inference for hawkes processes,” *Methodology and Computing in Applied Probability*, vol. 15, no. 3, pp. 623–642, 2013.
- [34] K. Zhou, H. Zha, and L. Song, “Learning social infectivity in sparse low-rank networks using multi-dimensional hawkes processes,” in *AISTATS*, 2013.
- [35] E. C. Hall and R. M. Willett, “Tracking dynamic point processes on networks,” *Transactions on Information Theory*, vol. 62, no. 7, pp. 4327–4346, 2016.
- [36] E. Lewis and G. Mohler, “A nonparametric em algorithm for multiscale hawkes processes,” *Journal of Nonparametric Statistics*, 2011.
- [37] P. Reynaud-Bouret, S. Schbath, *et al.*, “Adaptive estimation for hawkes processes; application to genome analysis,” *The Annals of Statistics*, vol. 38, no. 5, pp. 2781–2822, 2010.
- [38] N. R. Hansen, P. Reynaud-Bouret, V. Rivoirard, *et al.*, “Lasso and probabilistic inequalities for multivariate point processes,” *Bernoulli*, vol. 21, no. 1, pp. 83–143, 2015.

- [39] M. Eichler, R. Dahlhaus, and J. Dueck, “Graphical modeling for multivariate hawkes processes with nonparametric link functions,” *Journal of Time Series Analysis*, 2016.
- [40] E. Bacry, K. Dayri, and J.-F. Muzy, “Non-parametric kernel estimation for symmetric hawkes processes. application to high frequency financial data,” *The European Physical Journal B*, vol. 85, no. 5, pp. 1–12, 2012.
- [41] N. Du, L. Song, M. Yuan, and A. J. Smola, “Learning networks of heterogeneous influence,” in *NIPS*, 2012.
- [42] R. Lemonnier and N. Vayatis, “Nonparametric markovian learning of triggering kernels for mutually exciting and mutually inhibiting multivariate hawkes processes,” in *ECML-KDD*, 2014.
- [43] R. Kobayashi and R. Lambiotte, “Tideh: Time-dependent hawkes process for predicting retweet dynamics,” *ArXiv preprint arXiv:1603.09449*, 2016.
- [44] H. Mei and J. Eisner, “The neural hawkes process: A neurally self-modulating multivariate point process,” *ArXiv preprint arXiv:1612.09328*, 2016.
- [45] K. Chwialkowski and A. Gretton, “A kernel independence test for random processes,” in *ICML*, 2014.
- [46] A. Gunawardana, C. Meek, and P. Xu, “A model for temporal dependencies in event streams,” in *NIPS*, 2011.
- [47] D. Song, H. Wang, C. Y. Tu, V. Z. Marmarelis, R. E. Hampson, S. A. Deadwyler, and T. W. Berger, “Identification of sparse neural functional connectivity using penalized likelihood estimation and basis functions,” *Journal of computational neuroscience*, vol. 35, no. 3, pp. 335–357, 2013.
- [48] H. Daneshmand, M. Gomez-Rodriguez, L. Song, and B. Schoelkopf, “Estimating diffusion network structures: Recovery conditions, sample complexity & soft-thresholding algorithm,” in *ICML*, 2014.
- [49] S. J. Yakowitz and J. D. Spragins, “On the identifiability of finite mixtures,” *The Annals of Mathematical Statistics*, pp. 209–214, 1968.
- [50] C. Bruni and G. Koch, “Identifiability of continuous mixtures of unknown gaussian distributions,” *The Annals of Probability*, pp. 1341–1357, 1985.
- [51] C. E. Rasmussen, “The infinite gaussian mixture model,” in *NIPS*, 1999.

- [52] C. Maugis, G. Celeux, and M.-L. Martin-Magniette, “Variable selection for clustering with gaussian mixture models,” *Biometrics*, vol. 65, no. 3, pp. 701–709, 2009.
- [53] D. Luo, H. Xu, Y. Zhen, B. Dilkina, H. Zha, X. Yang, and W. Zhang, “Learning mixtures of markov chains from aggregate data with structural constraints,” *Transactions on Knowledge and Data Engineering*, vol. 28, no. 6, pp. 1518–1531, 2016.
- [54] D. M. Blei and M. I. Jordan, “Variational inference for dirichlet process mixtures,” *Bayesian analysis*, vol. 1, no. 1, pp. 121–143, 2006.
- [55] R. Socher, A. L. Maas, and C. D. Manning, “Spectral chinese restaurant processes: Nonparametric clustering based on similarities,” in *AISTATS*, 2011.
- [56] N. Du, M. Farajtabar, A. Ahmed, A. J. Smola, and L. Song, “Dirichlet-hawkes processes with applications to clustering continuous-time document streams,” in *SIGKDD*, 2015.
- [57] B. W. Turnbull, “Nonparametric estimation of a survivorship function with doubly censored data,” *Journal of the American Statistical Association*, vol. 69, no. 345, pp. 169–173, 1974.
- [58] V. De Gruttola and S. W. Lagakos, “Analysis of doubly-censored survival data, with application to aids,” *Biometrics*, pp. 1–11, 1989.
- [59] J. P. Klein and M. L. Moeschberger, *Survival analysis: Techniques for censored and truncated data*. Springer Science & Business Media, 2005.
- [60] G. J. Van den Berg and B. Drepper, “Inference for shared-frailty survival models with left-truncated data,” *Econometric Reviews*, vol. 35, no. 6, pp. 1075–1098, 2016.
- [61] R. L. Streit, *Poisson point processes: Imaging, tracking, and sensing*. Springer Science & Business Media, 2010.
- [62] C.-P. S. Fan, “Local likelihood for interval-censored and aggregated point process data,” PhD thesis, University of Toronto, 2009.
- [63] J. Sun and J. Kalbfleisch, “Estimation of the mean function of point processes based on panel count data,” *Statistica Sinica*, pp. 279–289, 1995.
- [64] J. A. Wellner and Y. Zhang, “Two estimators of the mean of a counting process with panel count data,” *Annals of Statistics*, pp. 779–814, 2000.
- [65] D. B. Rubin, *Multiple Imputation for Nonresponse in Surveys*. John Wiley & Sons, 2009, vol. 307.

- [66] B. Efron, *The jackknife, the bootstrap and other resampling plans*. SIAM, 1982, vol. 38.
- [67] H. R. Kunsch, “The jackknife and the bootstrap for general stationary observations,” *The Annals of Statistics*, pp. 1217–1241, 1989.
- [68] D. N. Politis and J. P. Romano, “The stationary bootstrap,” *Journal of the American Statistical association*, vol. 89, no. 428, pp. 1303–1313, 1994.
- [69] S. Gonçalves and L. Kilian, “Bootstrapping autoregressions with conditional heteroskedasticity of unknown form,” *Journal of Econometrics*, vol. 123, no. 1, pp. 89–120, 2004.
- [70] E. Paparoditis and D. N. Politis, “Tapered block bootstrap,” *Biometrika*, vol. 88, no. 4, pp. 1105–1119, 2001.
- [71] Y. Guan and J. M. Loh, “A thinned block bootstrap variance estimation procedure for inhomogeneous spatial point patterns,” *Journal of the American Statistical Association*, vol. 102, no. 480, pp. 1377–1386, 2007.
- [72] A. Cowling, P. Hall, and M. J. Phillips, “Bootstrap confidence regions for the intensity of a poisson point process,” *Journal of the American Statistical Association*, vol. 91, no. 436, pp. 1516–1524, 1996.
- [73] P. D. Kirk and M. P. Stumpf, “Gaussian process regression bootstrapping: Exploring the effects of uncertainty in time course data,” *Bioinformatics*, vol. 25, no. 10, pp. 1300–1306, 2009.
- [74] I. K. Sethi and G. Sarvarayudu, “Hierarchical classifier design using mutual information,” *Transactions on Pattern Analysis and Machine Intelligence*, no. 4, pp. 441–445, 1982.
- [75] N. T. Longford, “A fast scoring algorithm for maximum likelihood estimation in unbalanced mixed models with nested random effects,” *Biometrika*, vol. 74, no. 4, pp. 817–827, 1987.
- [76] G. King and L. Zeng, “Logistic regression in rare events data,” *Political Analysis*, pp. 137–163, 2001.
- [77] S. Tan, “Neighbor-weighted k-nearest neighbor for unbalanced text corpus,” *Expert Systems with Applications*, vol. 28, no. 4, pp. 667–671, 2005.
- [78] D. Zhuang, B. Zhang, Q. Yang, J. Yan, Z. Chen, and Y. Chen, “Efficient text classification by weighted proximal svm,” in *ICDM*, IEEE, 2005, pp. 8–15.

- [79] W. S. Lee and B. Liu, “Learning with positive and unlabeled examples using weighted logistic regression,” in *ICML*, vol. 3, 2003, pp. 448–455.
- [80] H. Xu, L. Yu, D. Luo, H. Zha, and Y. Xu, “Dictionary learning with mutually reinforcing group-graph structures,” in *AAAI*, 2015.
- [81] H. Xu and H. Zha, “Manifold based face synthesis from sparse samples,” in *ICCV*, 2013.
- [82] H. Xu, H. Zha, and M. Davenport, “Manifold based dynamic texture synthesis from extremely few samples,” in *CVPR*, 2014.
- [83] H. Yang, Z. Xu, I. King, and M. R. Lyu, “Online learning for group lasso,” in *ICML*, 2010.
- [84] N. Simon, J. Friedman, T. Hastie, and R. Tibshirani, “A sparse-group lasso,” *Journal of Computational and Graphical Statistics*, vol. 22, no. 2, pp. 231–245, 2013.
- [85] V. O. Alan, W. S. Ronald, and R. John, “Discrete-time signal processing,” *New Jersey, Printice Hall Inc*, 1989.
- [86] B. W. Silverman, *Density estimation for statistics and data analysis*. CRC press, 1986, vol. 26.
- [87] D. Luo, H. Xu, H. Zha, J. Du, R. Xie, X. Yang, and W. Zhang, “You are what you watch and when you watch: Inferring household structures from iptv viewing data,” *Transactions on Broadcasting*, vol. 60, no. 1, pp. 61–72, 2014.
- [88] A Evgeniou and M. Pontil, “Multi-task feature learning,” in *NIPS*, 2007.
- [89] J. Liu, S. Ji, and J. Ye, “Multi-task feature learning via efficient l_2 , l_1 -norm minimization,” in *UAI*, 2009.
- [90] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, “Distributed optimization and statistical learning via the alternating direction method of multipliers,” *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.
- [91] H. Ouyang, N. He, L. Tran, and A. Gray, “Stochastic alternating direction method of multipliers,” in *ICML*, 2013.
- [92] H. Zha, X. He, C. Ding, M. Gu, and H. D. Simon, “Spectral relaxation for k-means clustering,” in *NIPS*, 2001.
- [93] A. Y. Ng, M. I. Jordan, Y. Weiss, *et al.*, “On spectral clustering: Analysis and an algorithm,” in *NIPS*, 2002.

- [94] B. G. Lindsay, “Mixture models: Theory, geometry and applications,” in *NSF-CBMS regional conference series in probability and statistics*, 1995.
- [95] D. Kim, “Mixture inference at the edge of identifiability,” *Ph.D. Thesis*, 2008.
- [96] L. Li and H. Zha, “Dyadic event attribution in social networks with mixtures of hawkes processes,” in *CIKM*, 2013.
- [97] Y. W. Teh, D. Newman, and M. Welling, “A collapsed variational bayesian inference algorithm for latent dirichlet allocation,” in *NIPS*, 2006.
- [98] P. J. Green, “Reversible jump markov chain monte carlo computation and bayesian model determination,” *Biometrika*, pp. 711–732, 1995.
- [99] Z. Zhang, K. L. Chan, Y. Wu, and C. Chen, “Learning a multivariate gaussian mixture model with the reversible jump mcmc algorithm,” *Statistics and Computing*, vol. 14, no. 4, pp. 343–355, 2004.
- [100] G. H. Golub, Z. Zhang, and H. Zha, “Large sparse symmetric eigenvalue problems with homogeneous linear constraints: The lanczos process with inner–outer iterations,” *Linear Algebra And Its Applications*, vol. 309, no. 1, pp. 289–306, 2000.
- [101] B. A. Ogunnaike and W. H. Ray, *Process dynamics, modeling, and control*. Oxford University Press, USA, 1994.
- [102] J. Snoek, H. Larochelle, and R. P. Adams, “Practical bayesian optimization of machine learning algorithms,” in *NIPS*, 2012.
- [103] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, “Taking the human out of the loop: A review of bayesian optimization,” *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016.
- [104] E. Meijer and J. Y. Ypma, “A simple identification proof for a mixture of two univariate normal distributions,” *Journal of Classification*, vol. 25, no. 1, pp. 113–123, 2008.
- [105] T. J. Rothenberg, “Identification in parametric models,” *Econometrica: Journal of the Econometric Society*, pp. 577–591, 1971.
- [106] D. Görür and C. E. Rasmussen, “Dirichlet process gaussian mixture models: Choice of the base distribution,” *Journal of Computer Science and Technology*, vol. 25, no. 4, pp. 653–664, 2010.
- [107] R. Tibshirani and G. Walther, “Cluster validation by prediction strength,” *Journal of Computational and Graphical Statistics*, vol. 14, no. 3, pp. 511–528, 2005.

- [108] U. Von Luxburg, *Clustering Stability*. Now Publishers Inc, 2010.
- [109] M. Saeed, C. Lieu, G. Raber, and R. G. Mark, “Mimic ii: A massive temporal icu patient database to support research in intelligent patient monitoring,” in *Computers in Cardiology*, 2002.
- [110] E. Choi, N. Du, R. Chen, L. Song, and J. Sun, “Constructing disease network and temporal progression model via context-sensitive hawkes process,” in *ICDM*, 2015.
- [111] A. E. Johnson, T. J. Pollard, L. Shen, L.-w. H. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark, “Mimic-iii, a freely accessible critical care database,” *Scientific data*, vol. 3, 2016.
- [112] R. A. Deyo, D. C. Cherkin, and M. A. Ciol, “Adapting a clinical comorbidity index for use with icd-9-cm administrative databases,” *Journal of clinical epidemiology*, vol. 45, no. 6, pp. 613–619, 1992.
- [113] I. of Medicine Committee on the Future of Emergency Care in the US Health System *et al.*, “The future of emergency care in the united states health system,” *Annals of emergency medicine*, vol. 48, no. 2, p. 115, 2006.
- [114] S. Trzeciak and E. Rivers, “Emergency department overcrowding in the united states: An emerging threat to patient safety and public health,” *Emergency medicine journal*, vol. 20, no. 5, pp. 402–405, 2003.
- [115] J. S. Olshaker, “Managing emergency department overcrowding,” *Emergency medicine clinics of North America*, vol. 27, no. 4, pp. 593–603, 2009.
- [116] J. L. Pascual, N. W. Blank, D. N. Holena, M. P. Robertson, M. Diop, S. R. Allen, N. D. Martin, B. A. Kohl, C. A. Sims, C. W. Schwab, *et al.*, “There s no place like home: Boarding surgical icu patients in other icus and the effect of distances from the home unit,” *The journal of trauma and acute care surgery*, vol. 76, no. 4, p. 1096, 2014.
- [117] P. H. Knight, N. Maheshwari, J. Hussain, M. Scholl, M. Hughes, T. J. Papadimos, W. A. Guo, J. Cipolla, S. P. Stawicki, and N. Latchana, “Complications during intrahospital transport of critically ill patients: Focus on risk identification and prevention,” *International Journal of Critical Illness and Injury Science*, vol. 5, no. 4, p. 256, 2015.
- [118] D. B. Chalfin, S. Trzeciak, A. Likourezos, B. M. Baumann, R. P. Dellinger, D.-E. study group, *et al.*, “Impact of delayed transfer of critically ill patients from the emergency department to the intensive care unit,” *Critical care medicine*, vol. 35, no. 6, pp. 1477–1483, 2007.

- [119] A Ziser, M Alkobi, R Markovits, and B Rozenberg, “The postanesthesia care unit as a temporary admission location due to intensive care and ward overflow,” *British journal of anaesthesia*, vol. 88, no. 4, pp. 577–579, 2002.
- [120] J. R. Dichter, R. K. Kanter, D. Dries, V. Luyckx, M. L. Lim, J. Wilgis, M. R. Anderson, B. Sarani, N. Hupert, R. Mutter, *et al.*, “System-level planning, coordination, and communication: Care of the critically ill and injured during pandemics and disasters: Chest consensus statement,” *CHEST Journal*, vol. 146, no. 4_suppl, e87S–e102S, 2014.
- [121] K. Wongsuphasawat and D. Gotz, “Outflow: Visualizing patient flow by symptoms and outcome,” in *IEEE VisWeek Workshop on Visual Analytics in Healthcare*, American Medical Informatics Association, 2011, pp. 25–28.
- [122] M. Vankipuram, K. Kahol, T. Cohen, and V. L. Patel, “Toward automated workflow analysis and visualization in clinical environments,” *Journal of biomedical informatics*, vol. 44, no. 3, pp. 432–440, 2011.
- [123] M. Blount, M. R. Ebling, J. M. Eklund, A. G. James, C. McGregor, N. Percival, K. P. Smith, and D. Sow, “Real-time analysis for intensive care: Development and deployment of the artemis analytic system,” *Engineering in Medicine and Biology Magazine*, vol. 29, no. 2, pp. 110–118, 2010.
- [124] G. Hripcsak and D. J. Albers, “Next-generation phenotyping of electronic health records,” *Journal of the American Medical Informatics Association*, vol. 20, no. 1, pp. 117–121, 2013.
- [125] C. Liu, F. Wang, J. Hu, and H. Xiong, “Temporal phenotyping from longitudinal electronic health records: A graph based framework,” in *SIGKDD*, ACM, 2015, pp. 705–714.
- [126] Y. Wang, R. Chen, J. Ghosh, J. C. Denny, A. Kho, Y. Chen, B. A. Malin, and J. Sun, “Rubik: Knowledge guided tensor factorization and completion for health data analytics,” in *SIGKDD*, ACM, 2015, pp. 1265–1274.
- [127] Z. Che, D. Kale, W. Li, M. T. Bahadori, and Y. Liu, “Deep computational phenotyping,” in *SIGKDD*, ACM, 2015, pp. 507–516.
- [128] M. Ozkaynak, O. Dziadkowiec, R. Mistry, T. Callahan, Z. He, S. Deakyne, and E. Tham, “Characterizing workflow for pediatric asthma patients in emergency departments using electronic health records,” *Journal of biomedical informatics*, vol. 57, pp. 386–398, 2015.
- [129] M. Ozkaynak, P. F. Brennan, D. A. Hanauer, S. Johnson, J. Aarts, K. Zheng, and S. N. Haque, “Patient-centered care requires a patient-oriented workflow model,”

Journal of the American Medical Informatics Association, vol. 20, no. e1, e14–e16, 2013.

- [130] R. Cegłowski, L. Churilov, and J. Wasserthiel, “Facilitating decision support in hospital emergency departments: A process oriented perspective,” *ECIS Proceedings*, p. 55, 2005.
- [131] A. S. Cegłowski, “An investigation of emergency department overcrowding using data mining and simulation: A patient treatment type perspective,” PhD thesis, Monash University. Faculty of Business, Economics. Department of Accounting, and Finance, 2006.
- [132] A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, “Physiobank, physiotoolkit, and physionet components of a new research resource for complex physiologic signals,” *Circulation*, vol. 101, no. 23, e215–e220, 2000.
- [133] Y. Yang, H. T. Shen, Z. Ma, Z. Huang, and X. Zhou, “ ℓ_2 , 1-norm regularized discriminative feature selection for unsupervised learning,” in *IJCAI*, 2011.
- [134] S. Raman, T. J. Fuchs, P. J. Wild, E. Dahl, and V. Roth, “The bayesian group-lasso for analyzing contingency tables,” in *Proceedings of the 26th Annual International Conference on Machine Learning*, ACM, 2009, pp. 881–888.
- [135] X. Xu, M. Ghosh, *et al.*, “Bayesian variable selection and estimation for group lasso,” *Bayesian Analysis*, vol. 10, no. 4, pp. 909–936, 2015.
- [136] D. Gabay and B. Mercier, “A dual algorithm for the solution of nonlinear variational problems via finite element approximation,” *Computers & Mathematics with Applications*, vol. 2, no. 1, pp. 17–40, 1976.
- [137] R. Nishihara, L. Lessard, B. Recht, A. Packard, and M. I. Jordan, “A general analysis of the convergence of admm,” *ArXiv preprint*, 2015.
- [138] S. Mosci, L. Rosasco, M. Santoro, A. Verri, and S. Villa, “Solving structured sparsity regularization with proximal methods,” in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, 2010, pp. 418–433.
- [139] N. Parikh, S. P. Boyd, *et al.*, “Proximal algorithms,” *Foundations and Trends in optimization*, vol. 1, no. 3, pp. 127–239, 2014.
- [140] W. J. Anderson, *Continuous-time Markov chains: An applications-oriented approach*. Springer Science & Business Media, 2012.

- [141] T. Schaul, S. Zhang, and Y. LeCun, “No more pesky learning rates,” in *ICML*, 2013, pp. 343–351.
- [142] Y. Gong and X. Liu, “Video summarization using singular value decomposition,” in *CVPR*, 2000.
- [143] Y. Li, T. Zhang, and D. Tretter, “An overview of video abstraction techniques,” *HP Laboratories Palo Alto*, 2001.
- [144] Y.-F. Ma, X.-S. Hua, L. Lu, and H.-J. Zhang, “A generic framework of user attention model and its application in video summarization,” *Transactions on Multimedia*, vol. 7, no. 5, pp. 907–919, 2005.
- [145] J. You, G. Liu, L. Sun, and H. Li, “A multiple visual models based perceptive analysis framework for multilevel video summarization,” *Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 3, pp. 273–285, 2007.
- [146] Y. Li, S.-H. Lee, C.-H. Yeh, and C.-C. Kuo, “Techniques for movie content analysis and skimming: Tutorial and overview on video abstraction techniques,” *Signal Processing Magazine*, vol. 23, no. 2, pp. 79–89, 2006.
- [147] L. Li, K. Zhou, G.-R. Xue, H. Zha, and Y. Yu, “Video summarization via transferable structured learning,” in *WWW*, 2011.
- [148] W. Jiang, C. Cotton, and A. C. Loui, “Automatic consumer video summarization by audio and visual analysis,” in *ICME*, 2011.
- [149] G. Evangelopoulos, A. Zlatintsi, A. Potamianos, P. Maragos, K. Rapantzikos, G. Skoumas, and Y. Avrithis, “Multimodal saliency and fusion for movie summarization based on aural, visual, and textual attention,” *Transactions on Multimedia*, vol. 15, no. 7, pp. 1553–1568, 2013.
- [150] J. Yan, M. Zhu, H. Liu, and Y. Liu, “Visual saliency detection via sparsity pursuit,” *Signal Processing Letters*, vol. 17, no. 8, pp. 739–742, 2010.
- [151] Z. Wang, M. Kumar, J. Luo, and B. Li, “Sequence-kernel based sparse representation for amateur video summarization,” in *ACM workshop on Modeling and representing events*, 2011.
- [152] M. Wang, R. Hong, G. Li, Z.-J. Zha, S. Yan, and T.-S. Chua, “Event driven web video summarization by tag localization and key-shot identification,” *Transactions on Multimedia*, vol. 14, no. 4, pp. 975–985, 2012.
- [153] Y. J. Lee, J. Ghosh, and K. Grauman, “Discovering important people and objects for egocentric video summarization,” in *CVPR*, 2012.

- [154] A. Khosla, R. Hamid, C.-J. Lin, and N. Sundaresan, “Large-scale video summarization using web-image priors,” in *CVPR*, 2013.
- [155] G. Kim, L. Sigal, and E. P. Xing, “Joint summarization of large-scale collections of web images and videos for storyline reconstruction,” in *CVPR*, 2014.
- [156] Y.-F. Ma, L. Lu, H.-J. Zhang, and M. Li, “A user attention model for video summarization,” in *ACM Multimedia*, 2002.
- [157] T. Hospedales, S. Gong, and T. Xiang, “A markov clustering topic model for mining behaviour in video,” in *ICCV*, 2009.
- [158] G. Irie, K. Hidaka, T. Satou, A. Kojima, T. Yamasaki, and K. Aizawa, “Latent topic driving model for movie affective scene classification,” in *ACM Multimedia*, 2009.
- [159] L. Itti and P. Baldi, “Bayesian surprise attracts human attention,” *Vision research*, vol. 49, no. 10, pp. 1295–1306, 2009.
- [160] G. Irie, T. Satou, A. Kojima, T. Yamasaki, and K. Aizawa, “Automatic trailer generation,” in *ACM Multimedia*, 2010.
- [161] X. Hou, J. Harel, and C. Koch, “Image signature: Highlighting sparse salient regions,” *Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 1, pp. 194–201, 2012.
- [162] E. W. Dijkstra, “A note on two problems in connexion with graphs,” *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.
- [163] Y. Xu, P. Müller, and D. Telesca, “Bayesian inference for latent biologic structure with determinantal point processes (dpp),” *Biometrics*, 2016.
- [164] A. Simma and M. I. Jordan, “Modeling events with cascades of poisson processes,” in *UAI*, 2010.

VITA

Hongteng Xu is a Ph.D. candidate in School of Electrical and Computer Engineering, Georgia Tech. His primary research interest is machine learning and its applications of data mining and computer vision, especially point process models and learning algorithms for synchronous/asynchronous event sequences analysis and prediction. His work has inspired broad potential applications, e.g., social behavior analysis, medical informatics, visual attractiveness modeling. Hongteng has received several achievements for his research, including over 10 publications on top conferences (ICML, IJCAI, AAAI, CVPR, ICCV) and journals (TKDE, TIP, TPAMI), traveling award of ICCV 2013 and ICML 2016, several invited talks at companies, research institutes, and universities, and entering the finalist of Baidu Fellowship 2016.